Security Risk Management for the Internet of Things

Technologies and Techniques for IoT Security, Privacy and Data Protection

John Soldatos
(Editor)

now
the essence of knowledge

Published, sold and distributed by:
now Publishers Inc.
PO Box 1024
Hanover, MA 02339
United States
Tel. +1-781-985-4510
www.nowpublishers.com
sales@nowpublishers.com

Outside North America:
now Publishers Inc.
PO Box 179
2600 AD Delft
The Netherlands
Tel. +31-6-51115274

ISBN: 978-1-68083-682-0
E-ISBN: 978-1-68083-683-7
DOI: 10.1561/9781680836837

Copyright © 2020 John Soldatos

Suggested citation: John Soldatos (ed.). (2020). *Security Risk Management for the Internet of Things*. Boston–Delft: Now Publishers

The work will be available online open access and governed by the Creative Commons "Attribution-Non Commercial" License (CC BY-NC), according to https://creativecommons.org/licenses/by-nc/4.0/

Table of Contents

Foreword xi

Preface xv

Glossary xxi

Chapter 1 Introduction 1

By John Soldatos

1.1 Introduction 1
1.2 Overview and Limitations of Security Risk Assessment Frameworks 5
1.2.1 Overview of Security Risk Assessment 5
1.2.2 Limitations of Security Risk Assessment Frameworks for IoT 7
1.3 New Technology Enablers and Novel Security Concepts 9
1.3.1 IoT Security Knowledge Bases 9
1.3.2 IoT Reference Architectures and Security Frameworks 10
1.3.3 Blockchain Technology for Decentralized Secure Data Sharing for Security in IoT Value Chains 10
1.3.4 Technologies Facilitating GDPR Compliance 12
1.3.5 Machine Learning and Artificial Intelligence Technologies for Data-driven Security 13
1.4 Conclusion 13
Acknowledgments 14
References 14

Chapter 2 Security Data Modelling for Configurable Risk Assessment as a Service in IoT Systems 17

By Nikos Kefalakis, Angela-Maria Despotopoulou, Spyridon Evangelatos and John Soldatos

2.1 Introduction 18
2.2 Data-driven Security Architecture 21
2.2.1 Overview 21
2.2.2 The Data Management Group 23
2.2.3 The Analytics Group 23
2.2.4 The Global Repository 24
2.2.5 The Security and Privacy Group 25
2.2.6 The Risk Assessment Service Group 26
2.2.7 The Compliance Auditing Service (CAS) Group 26
2.2.8 The Programming Support Group 27
2.2.9 The SLA Group 27
2.3 Data Modelling for Security Systems Interoperability and Configurability 28
2.3.1 Overview 28
2.3.2 Modelling Security Data: The Observation Entity 29
2.3.3 Configuring and Managing the Data Collection and Routing Process: The Data Management Group 30
2.3.4 Modelling Security Analytics: The Analytics Group 32
2.3.5 Security Knowledge Base 33
2.3.6 Modelling for Risk Assessment Services: The Risk Assessment Group 33
2.3.7 Configuring the Data-driven Security System: The Configuration Management Database 34
2.3.8 Managing Service Level Agreements (SLA): The SLA Group 35
2.4 Risk Assessment Services 36
2.4.1 Risk Assessment & Mitigation Service Overview and Components 36
2.4.2 Risk Assessment & Mitigation Implementation Scenario 38
2.4.3 Modelling of Security Information Flows and Reports 38
2.4.3.1 Risk Analysis 40
2.4.3.2 SecureIoT Required Configuration Entities 41
2.4.3.3 RA&MS-specific Configuration Entities 44
2.5 Conclusions 46
Acknowledgments 47
References 47

Chapter 3 Data-driven IoT Security Using Deep Learning Techniques 49

By Stefanos Astaras, Nikos Kefalakis, Angela-Maria Despotopoulou and John Soldatos

3.1 Introduction ... 50
3.2 Methodology and Datasets ... 53
3.2.1 CRISP-DM Methodology ... 53
3.2.2 Connected Cars Dataset ... 54
3.2.3 Socially Assistive Robots Datasets ... 54
3.3 Variational Autoencoders for Anomaly Detection ... 55
3.3.1 VAE Architecture ... 55
3.3.2 VAE Training ... 57
3.3.3 Algorithm Fitness ... 58
3.4 Application and Validation Results ... 59
3.4.1 Anomaly Detection in Connected Cars ... 59
3.4.2 Anomaly Detection in Socially Assistive Robots Use Cases ... 61
3.4.2.1 QT Robot Dataset ... 61
3.4.2.2 IoT-cloud Platform (CloudCare2U) Dataset ... 62
3.4.3 Prototype Implementation ... 65
3.5 Conclusions and Discussion ... 65
Acknowledgments ... 66
References ... 66

Chapter 4 Privacy Awareness, Risk Assessment, and Control Measures in IoT Platforms: BRAIN-IoT Approach 69

By Mohammad Rifat Ahmmad Rashid, Davide Conzon, Xu Tao and Enrico Ferrera

4.1 Introduction ... 70
4.2 Literature Review ... 71
4.2.1 GDPR Requirements Related to IoT Domain ... 71
4.2.2 Current Standards and Tools for PIA ... 72
4.3 A Conceptual Privacy Awareness Framework ... 74
4.3.1 Context ... 74
4.3.2 Privacy Principles ... 75
4.3.3 Privacy Risks ... 78
4.3.4 Privacy Compliance Evaluation ... 79
4.4 Experimental Analysis ... 80
4.5 Discussion ... 84
4.6 Conclusion and Future Work ... 85
Acknowledgments ... 86
References ... 86

Chapter 5 IoT Network Risk Assessment and Mitigation: The SerIoT Approach 88

By Gianmarco Baldini, Piotr Fröhlich, Erol Gelenbe, Jose Luis Hernandez-Ramos, Mateusz Nowak, Slawek Nowak, Stavros Papadopoulos, Anastasis Drosou and Dimitrios Tzovaras

5.1 Introduction 89
5.2 Risk Management in IoT 90
5.3 Autopolicy System 93
5.4 Towards Distributed Attack Detection 95
5.5 Conclusions 99
Acknowledgments 100
References 100

Chapter 6 Chariot-integrated Approach to Safety, Privacy, and Security – CHARIOT IPSE 105

By Aydin Ulas, Bora Caglayan, Sofiane Zemouri, George Theofilis, Konstantinos Loupos, Antonis Mygiakis, Andrea Battaglia, Mario Villiani, Christos Skoufis and Stelios Christofi

6.1 The CHARIOT Safety Supervision Engine 106
6.2 The CHARIOT Privacy Engine 110
6.2.1 IoTL Language Extension—Access Control 111
6.3 The CHARIOT Security Engine 114
6.3.1 Up-to-date Firmware 115
6.3.2 Firmware Threats and Exploitations 116
6.4 IPSE Dashboard and User Interfacing 116
6.5 Conclusions and Future Work 119
Acknowledgments 120
References 120

Chapter 7 Pattern-driven Security, Privacy, Dependability and Interoperability in IoT 121

By Nikolaos Petroulakis, Konstantinos Fysarakis, Henrich C. Pöhls, Vivek Kulkarni, George Spanoudakis, Arne Bröring, Manos Papoutsakis, Manolis Michalodimitrakis and Sotiris Ioannidis

7.1 Introduction 121
7.2 Background and Challenges 123
7.2.1 IoT Security 124
7.2.2 Privacy Invasion 124
7.2.3 Network Dependability 125

7.2.4 IoT Interoperability ... 127
7.2.5 Achieving Security, Privacy, Dependability and Interoperability by Design ... 128
7.3 SPDI Patterns ... 129
7.3.1 Pattern Language ... 130
7.3.2 Machine-processable Pattern Encoding ... 131
7.3.3 Reasoning Components ... 131
7.4 Pattern Enforcement's and Evaluation in SEMIoTICS Use Cases ... 133
7.4.1 Pattern-enabled IoT Orchestrations ... 133
7.4.2 Security and Privacy Policy Enforcement Patterns ... 133
7.4.3 Service Function Chaining and SPDI Patterns ... 135
7.5 Conclusion ... 137
Acknowledgments ... 137
References ... 137

Chapter 8 Enabling Continuous Privacy Risk Management in IoT Systems 143

By Victor Muntés-Mulero, Jacek Dominiak, Elena González and David Sanchez-Charles

8.1 Introduction ... 143
8.2 State of the Art ... 145
8.3 Model-based Continuous Risk Management Methodology ... 147
8.4 Automatic Vulnerability Detection ... 149
8.5 Model-based Risk Management Approach ... 153
8.6 Conclusions ... 157
Acknowledgments ... 158
References ... 158

Chapter 9 Data Protection Compliance Requirements for the Internet of Things 161

By Luca Bolognini, Sébastien Ziegler, Pasquale Annicchino, Francesco Capparelli and Alice Audino

9.1 Introduction ... 161
9.2 IoT and General Data Protection Regulation: Awareness as a Key Safeguarding Factor ... 162
9.2.1 Awareness and Data Protection ... 162
9.2.2 Awareness of Data Subject as an Instrument for Opt-in and Free Choices ... 165
9.2.3 IoT Features Enabling Opting-out and Exercise of Data Subjects' Rights ... 167

9.3 The Principles of Accountability, Data Protection by Design and by Default as Indirect Requirements for IoT Technology Producers and Controllers . . . 170
9.3.1 Controllers vs Producers in IoT: Direct and Indirect GDPR Requirements . . . 170
9.3.2 Cybersecurity Measures for IoT: Recommendations . . . 173
9.3.3 Data Protection by Design Measures for IoT: Recommendations . . . 175
Conclusion and Acknowledgments . . . 177

Chapter 10 Cybersecurity Certification in IoT Environments 178
By Sara N. Matheu and Antonio F. Skarmeta

10.1 Introduction . . . 178
10.2 Security Certification Challenges in Current Schemes . . . 180
10.2.1 Current Security Certification Schemes . . . 182
10.3 The Two Perspectives of the ETSI Approach for Security Risk Assessment . . . 184
10.4 Proposed Approach for a Cybersecurity Certification Framework . . . 186
10.4.1 Establishing the Context . . . 188
10.4.2 Security Testing . . . 188
10.4.3 Security Risk Assessment . . . 190
10.4.4 Communicate and Consult: Labeling . . . 191
10.4.5 Monitoring and Review . . . 191
10.5 Conclusion . . . 192
Acknowledgments . . . 192
References . . . 192

Chapter 11 Firmware Software Analysis at Source Code and Binary Levels 196
By Franck Vedrine, Florent Kirchner, Basile Starynkevitch, Andrea Battaglia, Mario Villiani and Konstantinos Loupos

11.1 Scope, Business Orientation, and Purpose . . . 198
11.2 Technological Innovation and Security Alignment Per Outcome . . . 199
11.2.1 Securing Firmware Through Rule-based Code Analysis and Injection of Analysis Results and the Source Code Hash Within the Binary Code . . . 199
11.2.2 Securing Firmware During the Development Process . . . 200
11.3 Conclusions and Future Work . . . 206
Acknowledgments . . . 207
References . . . 207

Chapter 12 End-to-End Security for IoT 208
By Paul-Emmanuel Brun and Guillemette Massot
12.1 Introduction . . . 209
12.2 IoT Systems Architecture and Security Challenges . . . 209
12.3 State-of-the-Art Mitigation Measures . . . 212
12.3.1 Device Management . . . 213
12.3.2 Endpoint Protection . . . 213
12.3.3 Communication Security . . . 213
12.4 Bring End-to-End Security Layer to Low-power IoT Devices . . . 215
12.4.1 Strong Authentication of Devices . . . 215
12.4.2 End-to-End Encryption Data . . . 216
12.4.3 Optimized Key Management . . . 217
12.5 Conclusion . . . 218
Acknowledgments . . . 219

Chapter 13 Blockchain Ledger Solution Affirming Physical, Operational, and Functional Changes in an IoT System 220
By Alexandros Papageorgiou, Konstantinos Loupos and Thomas Krousarlis
13.1 CHARIOT Blockchain Requirements . . . 222
13.2 CHARIOT Technology Stack . . . 223
13.2.1 Administrator Keypair Manager . . . 223
13.2.2 RESTful API . . . 224
13.2.3 New Genesis Transactions . . . 224
13.2.4 Cryptographic Implementation . . . 225
13.3 Scalability, Integration, and Testing . . . 226
13.4 Conclusions and Future Work . . . 227
Acknowledgments . . . 227
References . . . 228

Chapter 14 Leveraging Interledger Technologies in IoT Security Risk Management 229
By Dmitrij Lagutin, Yki Kortesniemi, Vasilios A. Siris, Nikos Fotiou, George C. Polyzos and Lei Wu
14.1 Introduction . . . 229
14.2 Background . . . 232
14.2.1 Distributed Ledger Technologies (DLTs) . . . 232
14.2.2 Smart Contracts and Chaincode . . . 233
14.2.3 Interledger Functionality . . . 233
14.2.4 Decentralized Identifiers (DIDs) . . . 234
14.3 Previous Work . . . 235

14.4 Automated Responsible Disclosure (ARD) 237
14.5 Analysis 241
14.6 Conclusions and Future Work 243
Acknowledgments 244
References 244

Epilogue 247

Index 250

About the Editor 252

Contributing Authors 253

Foreword

Over twenty years following the introduction of the term Internet of Things (IoT) by Kevin Ashton at MIT, IoT technologies are growing at a rapid pace. In-line with early visions about IoT applications, IoT technologies are currently enabling advanced services that interconnect physical and virtual things based on interoperable communication technologies. Most importantly, IoT applications are no longer limited to small scale laboratory environments, as they are part of robust enterprise deployments. Consumers are already seeing IoT applications improving their lives and making the world a better place to live. For instance, thousands of internet-connected devices are currently optimizing resource usage and improving environmental performance in modern smart cities. As another example, IoT is a key enabler of advanced automation in industrial applications in sectors like manufacturing, energy, and supply chain management. During the past weeks we are also witnessing the importance of IoT technologies in fighting the COVID19 outbreak, through facilitating processes like diagnostic testing, contact tracing and disease spreading estimation.

These developments are however the start of the IoT journey rather than its destination. In several cases, state of the art IoT deployments have just scratched the surface of potential use cases and there is still a long way to go to realize the full potential of the IoT paradigm. The latter will be shaped by recent advances in technologies like Artificial Intelligence (AI), big data, robotics, blockchain and 5G. These technologies are expected to enable an even broader and progressing revolution in the future, across a wide variety of verticals, including healthcare, smart cities, energy, agriculture, and industry. Specifically, they will enable next generation IoT applications that will employ advanced and distributed computing to bring intelligence and automation at the point of action. Emerging IoT systems like connected and autonomous vehicles will be therefore able to take faster, more

intelligent, and more efficient decisions at the right place and at the right point in time.

In this landscape, the number of IoT devices is rising constantly with an expected 40 billion IoT devices to be in used worldwide by 2025[1], including not only passive devices, but also semi-autonomous smart objects. Nevertheless, the rising sophistication of IoT systems, technologies and applications is becoming associated with a more complex IoT ecosystem. This ecosystem is characterised by a combination of very diverse products, systems, and services, which rely on data and connectivity to deliver their value.

Delivering real value in the scope of the complex IoT ecosystem is primarily about addressing specific and pragmatic challenges, as needed for delivering applications that can be integrated into society. These applications must be part of an open, predictable, and competitive IoT market where individual rights and freedoms are respected. When it comes to addressing pragmatic challenges, the cybersecurity aspects of IoT come into the foreground. Specifically, security, privacy and data protection aspects cannot be negotiated when it comes to deploying real-life IoT applications that are ethical, trustworthy and protect the citizens' rights. In this context, the rising complexity of IoT technologies and ecosystems puts into question conventional methodologies for risk assessment and traditional regulatory paradigms. In fact, IoT applications are characterised by a very large attack surface and complex ecosystem consisting of diverse actors, heterogeneous physical and virtual spaces, as well as devices of different size, nature, and complexity. In this multi-actor, multi-stakeholder environment, cyber risks can mean different things to different actors, and hence, it is not always possible to have a single approach for confronting them. Furthermore, the complexity and fragmentation of applicable legal frameworks for safety and liability, at both EU and national levels, proves to be another significant challenge. Advanced digital technologies can become part of the solutions to these challenges, as they provide powerful testing and validation capabilities for managing cyber-security in complex environments.

The development of secure, trustworthy, and human-centric IoT systems and applications is a top priority for the European Commission. A human-centric and trustworthy approach is necessary to facilitate the uptake of digital solutions among European citizens. This approach reflects the EU's specific way and vision of human progress and evolution. Ensuring a cybersecure IoT is an essential foundation of this approach and vision. In Europe, any IoT solution must be compliant with the

1. https://www.statista.com/statistics/802690/worldwide-connected-devices-by-access-technology/

unique European Regulatory Framework, which has already a global impact on citizens and businesses. Key pillars of this framework are the E-privacy Directive 2002/58/EC, the General Data Protection Regulation 2016/679, and the Cybersecurity Regulation 2019/881. These directives and regulations have represented fundamental legal milestones ensuring that privacy and security are reinforced. Furthermore, under the Commission's new digital strategy, additional regulatory actions have been planned, notably the creation of a specific AI framework addressing safety and ethical challenges, as well as the adaptation of existing safety and liability frameworks to new technologies. Moreover, certification schemes for IoT privacy and security remain important regulatory options under the GDPR and Cybersecurity Act. The ultimate objective of past and ongoing work is to boost the development of cybersecure IoT environments, while ensuring that all pieces of relevant EU legislation and initiatives stay consistent among them and duplications are avoided.

For nearly fifteen years, the European Commission has been funding excellent research projects and innovation initiatives that have played a significant role in building the European IoT ecosystem. These have included several large-scale pilot projects, where IoT solutions were validated in real life settings. It has also supported a set of research projects in IoT security, including the projects that contribute to this book. Furthermore, the European Commission has encouraged and supported the clustering between projects, as a means of boosting their cooperation and facilitating them to share best practices with respect to fundamental horizontal topics such as security and privacy. Over the years, it has been proven that knowledge sharing, and collaboration provides added-value and leads to multiplicative benefits for the cooperating projects. The present book is an important outcome of the collaboration of nine European Commission funded projects on IoT security, which have collaborated efficiently and managed to identify common solutions beyond the specificities of individual projects. The nine projects have collaborated in inspiring and productive ways, in the scope of the H2020 IoT projects Security and Privacy Cluster, which has been established by in 2018.

As the European Commission officials in charge of the Activities of the Cluster, it is with great pleasure and satisfaction that we witness the completion of this book on IoT cybersecurity solutions, which emphasizes the projects' outcomes in relation to technologies and methodologies for IoT security risk management. The Commission welcomes this new book and its contribution to exploring innovative technical solutions on IoT security, as well as their relation to ongoing regulatory developments. We perceive the contents of the book as an invaluable contribution towards the ambition of building a "Thriving IoT ecosystem" underpinned by a

"Human Centred" approach. It is also very positive that this book is offered as an Open Access publication, which could help it reach a wider readership and boost its overall impact. Finally, it is our hope and expectation that this book will be proven an effective resource and a good reading experience for the IoT community.

June 2020

Salvatore Scalzo
EC Policy Officer, "Internet of Things" Unit of DG CONNECT
IoT Security and Privacy Cluster Coordinator

Franck Boissiere
EC Programme Officer, "Internet of Things" Unit of DG CONNECT
IoT Security and Privacy Cluster Coordinator

Preface

In recent years, Internet of Things (IoT) systems have rapidly evolved in terms of functional and technological sophistication. Early small-scale sensing systems and wireless sensor networks have given their place to massively scalable, cloud-based systems that comprise many thousands of internet-connected objects. At the same time, the advent of edge computing has provided opportunities for advanced internet of things deployments that process and analyze data in real time, while closing the loop to the field and influencing the status of the physical world. Moreover, we are gradually witnessing the rise of smart objects with semi-autonomous behavior (e.g., drones, robots, automated guided vehicles), which intensify the decentralization and the intelligence of edge/cloud architectures.

This rising complexity of IoT systems provides opportunities for more automated and intelligent business applications. However, it also introduces new cybersecurity challenges such as new ways for conducting large-scale attacks and a wealth of vulnerabilities and risks at different parts of an IoT system such as smart objects, IoT networks, edge gateways, and cloud elements. These vulnerabilities are evident in the scope of recent large-scale cybersecurity incidents, such as the Mirai malware that turned IoT devices into remotely controlled bots able to launch Distributed Denial of Service (DDoS) attacks. The Mirai-based based DDoS back in 2016 took advantage of vulnerabilities (e.g., hard-coded passwords, poorly patched software) of internet-connected CCTV (Closed Circuit Television) cameras and DVR (Digital Video Recorders). As another example, the "Lizard Stressor" attacks few years ago compromised many commercial home routers at a large scale.

In addition to confronting such attacks, developers, and operators of IoT systems must comply with stringent regulatory requirements, such as requirements

stemming from the General Data Protection Regulation (GDPR) and the Network Information Systems (NIS) directive. To this end, there is a need for deploying effective security and data protection methods that boost compliance to the security, privacy, and data protection mandates of these regulations.

In this context, security risk management methods can be a powerful tool for developers, solution integrators, and operators of IoT systems. International standards and frameworks for risk management can be used to support the identification of risks or threats, and to assess their respective probabilities. Nevertheless, state-of-the-art technologies for security risk assessment have prominent limitations when it comes to confronting risks associated with large-scale, cyber-physical, and interconnected IoT systems. For example, risk assessments for modern IoT systems must be more frequent and must take into account knowledge about both cyber and physical assets. Furthermore, they should be more proactive and automated, and able to leverage information that is shared across IoT supply chains. In this direction, organizations can take advantage of emerging technologies (e.g., edge/fog computing architectures, machine learning, blockchain), to implement novel risk assessment approaches that are characterized by automation, intelligence, and transparency.

During the last couple of years, a group of European Commission (EC) funded projects have been researching and implementing novel security and privacy mechanisms for the Internet of Things, including risk assessment and mitigation. The projects have formed a Cluster of IoT Security and Privacy projects, as a means of boosting their collaboration. The purpose of the present book is to present detailed information about the novel risk assessment techniques that have been developed by these projects and their role in the IoT security risk management process. Specifically, the book presents several architectures and platforms for end-to-end security, including their implementation based on the edge/fog computing paradigm. It also highlights machine learning techniques that boost the automation and proactiveness of IoT security risk assessments. Furthermore, blockchain solutions for open and transparent sharing of IoT security information across the supply chain are introduced. Moreover, several chapters of the book present frameworks for privacy awareness, along with technical measures that enable privacy risk assessment and boost GDPR compliance. Likewise, techniques for security certification of IoT systems along with frameworks for IoT security interoperability towards end-to-end security are discussed.

Overall, the book is structured into fifteen chapters that present different IoT security technologies, including novel techniques for IoT security risk assessment. Specifically:

- **Chapter 1 ("Introduction")** provides a brief overview of the main security challenges for modern IoT systems. It also introduces some of the most popular risk assessment frameworks and discusses their limitations. Furthermore, it illustrates some novel research directions in the area of cybersecurity for IoT systems, notably directions associated with security risk management.
- **Chapter 2 ("Security Data Modeling for Configurable Risk Assessment as a Service in IoT Systems")** introduces a data model that enables the development and configuration of data-driven IoT security systems, including systems that provide IoT Risk Assessment reports as a service. The introduced model has the virtue of combining security data about the status of the IoT system, with metadata about its configuration. The processing of the security data can provide insights on vulnerabilities, risk, and threats associated with IoT assets, as well as on mechanisms for delivering cyber-threat intelligence. At the same time, the management of the configuration metadata can be used for configuring data-driven cyber-threat intelligence functions for IoT systems. The model has been developed in the scope of the H2020 SecureIoT project, where it has been also validated in various IoT use cases in the areas of smart connected transport, smart manufacturing, and ambient assisted living.
- **Chapter 3 ("Data-driven IoT Security Using Deep Learning Techniques")** presents the use of deep learning techniques, and more specifically, Variational Autoencoder (VAE) techniques for anomaly detection in the behavior of smart objects like connected cars and socially assistive robots. It also illustrates the validation of these models over IoT datasets collected from these smart objects. The chapter discusses the advantages and limitations of using deep learning techniques for detecting security-related anomalies in the behavior of smart objects.
- **Chapter 4 ("Privacy Awareness, Risk Assessment, and Control Measures in IoT Platforms: BRAIN-IoT Approach")** presents an approach to embedding privacy awareness and privacy control features in IoT solutions. The presented approach has been developed in the scope of the H2020 Brain-IoT project and emphasizes the importance of privacy awareness in IoT systems in-line with the GDPR regulation. Furthermore, a conceptual framework for Privacy Impact Assessment (PIA) in-line with the privacy principles of the GDPR is presented as well.
- **Chapter 5 ("IoT Network Risk Assessment and Mitigation: The SerIoT Approach")** describes the IoT security risk assessment approach of the H2020 SerIoT project. First, it discusses reference IoT risk scenarios and reviews relevant mitigation techniques. Accordingly, a lightweight policy-based approach for risk mitigation at the level of the attachment

of IoT devices to a network is presented. The merits of a distributed Machine Learning (ML) approach to real-time risk and attack detection are discussed. Likewise, the chapter presents an attack mitigation approach that leverages adaptive ML to re-route traffic in IoT networks and to avoid compromised sections of the network. The chapter ends up presenting experimental results from SerIoT, along with suggestions for future work.

- **Chapter 6 ("CHARIOT Integrated Approach to Safety, Privacy, and Security")** focuses on the outcomes of the H2020 CHARIOT project. Specifically, it summarizes the platform design, analytics, and services developed to protect industrial data security, privacy, and safety. It also includes information about the architecture of the platform and the design of its services, as part of a decentralized, fog computing architecture and a dynamic IoT Safety Supervision Engine.
- **Chapter 7 ("Pattern-driven Security, Privacy, Dependability and Interoperability in IoT")** presents the development and evaluation of a pattern-driven approach for guaranteeing Security, Privacy, Dependability and Interoperability (SPDI) properties in the IoT domain. The chapter discusses how SPDI patterns can guarantee multi-layer end-to-end properties and how the enforcement of the patterns can help satisfy network dependability guarantees. Furthermore, the chapter illustrates how patterns can offer a solution to IoT semantic interoperability. The chapter is based on work carried out in the scope of the H2020 SEMIoTICS project.
- **Chapter 8 ("Enabling Continuous Privacy Risk Management in IoT Systems")** is devoted to studying the interactions between GDPR and the LINDDUN privacy threat modeling methodology. It also explores the connection between LINDDUN (Linkability, Identifiability, Non-repudiation, Detectability, Disclosure of information, Unawareness, and Non-compliance) and the STRIDE (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege) threat model towards enabling continuous evidence-based risk management and improving risk assessment for data protection. The chapter was contributed by the H2020 ENACT and PDP4E projects.
- **Chapter 9 ("Data Protection Compliance Assessment for the Internet of Things")** provides an overview of several methodologies and approaches, which ensure that IoT deployments comply with applicable data protection regulations. The chapter takes a practical approach based on the lessons learnt from activities that aimed at ensuring the compliance of various H2020 IoT Large Scale Pilot (LSP) projects with the GDPR regulation.

- **Chapter 10 ("Cybersecurity Certification in IoT Environments")** presents and analyzes current standardized IoT security certification approaches and outlines their gaps in terms of security certification (e.g., their limitations in dealing with dynamic security environments and/or security life cycle management). It also presents a methodology that combines security assessment and testing in order to address some of the challenges and to serve as a basis for future security certification schemes.
- **Chapter 11 ("Firmware Software Analysis at Source Code and Binary Levels")** presents recent research developments on the source code (precompilation) and binary (executable level) of firmware developed for IoT devices (sensors, gateways, and other nodes) in the H2020 CHARIOT project. It includes methodologies to capture source code or binary level vulnerabilities or malicious code, which may increase security risks and compromise the secure operation of IoT devices.
- **Chapter 12 ("End-to-end Security for IoT")** presents an approach for end-to-end protection and privacy for IoT systems. The approach covers all IoT assets from the device up to the application level. The approach was developed in the scope of the H2020 Brain-IoT project. It provides end-to-end integrity, encryption, and devices management.
- **Chapter 13 ("Blockchain Ledger Solution Affirming Physical, Operational, and Functional Changes in an IoT System")** introduces a blockchain solution for ensuring data legitimacy, trust, and privacy in an industrial IoT network. The chapter presents both the distributed ledger employed and the methodology followed. Specifically, it presents a design methodology and network setup compiled around an overall cognitive architecture that was developed in the scope of the H2020 CHARIOT project.
- **Chapter 14 ("Leveraging Interledger Technologies in IoT Security Risk Management")** introduces a novel approach to IoT security risk management based on the outcomes of the H2020 SOFIE project. The approach is based on the application of Distributed Ledger Technology (DLT) to securely and openly federate IoT platforms. Distributed ledgers provide a novel way to enable decentralized trust, while interledger technologies enable the bridging of multiple distributed ledgers together. This enables new opportunities for performing IoT security risk management in more open, flexible, and accountable way.
- **Chapter 15 ("Epilogue")** is the concluding chapter of the book. It summarizes the main takeaways from the previous chapters and provides an outlook for the future evolution of IoT security.

The target audience of the book includes:

- **Researchers in the area of IoT security**, who wish to be updated about latest and emerging IoT security trends, with emphasis on trends that shape the state-of-the-art IoT security risk management solutions.
- **IoT systems operators**, with an interest in adopting, deploying, and fully leveraging the next generation of IoT security technologies to secure their infrastructures and services.
- **Providers of IoT solutions and services**, who are interested in the implementation of use cases for the protection of their IoT assets.
- **Security experts and consultants** wishing to understand the future trends in IoT security towards devising solutions for securing the next generation of large-scale, massively decentralized IoT systems that comprise smart objects.

Overall, the book introduces novel IoT security technologies with emphasis on solutions for IoT risk assessment. It also presents future trends in IoT Security, including how novel digital technologies (e.g., Machine Learning, AI, blockchain) can be used to secure emerging IoT systems. Furthermore, it provides insights on privacy and data protection solutions, notably solutions that can support organizations in their regulatory compliance efforts. The book is made available as an Open Access publication, which could make it broadly and freely available to the IoT community. I would like to thank Now Publishers for the opportunity and their collaboration in making this happen.

Most importantly, I take the chance to thank the contributing projects for their valuable inputs and contributions in developing the presented IoT security technologies, as well as in documenting them as part of the book. I would also like to acknowledge funding and support from the European Commission as part of the H2020 SecureIoT, Brain-IoT, CHARIOT, ENACT, IoT Crawler, NGIOT, PDP4E, SEMIoTICS, SerIoT, and SOFIE projects.

March 2020
John Soldatos

Glossary

Symbols

2G - *Second Generation*. 209

5G - *Fifth Generation*. 121, 122

A

ABE - *Attribute-based Encryption*. 124

ACL - *Access Control Language*. 111

Adam - *Adaptive Moment Estimation*. 57

AES-CCM - *AES Cipher Counter Mode*. 216

AES-GCM - *AES Galois Counter Mode*. 216

AI - *Artificial Intelligence*. xx, 13, 19, 249

AIT - *Athens Information Technology*. 251

AMQP - *Advanced Message Queuing Protocol*. 127

ANN - *Artificial Neural Network*. 51

ANSSI - *Agence nationale de la sécurité des systèmes d'information*. 182

ANTLR - *ANother Tool for Language Recognition*. 108

API - *Application Programming Interfaces*. 109, 110, 116, 117, 128, 135, 197, 203, 224–226

ARD - *Automated Responsible Disclosure*. 229, 232, 234, 235, 237, 239, 243

ARM - *Advanced RISC (Reduced Instruction Set Computing) Machine*. 226

ASCII - *American Standard Code for Information Interchange*. 222

AVD - *Automated Vulnerability Detector*. 144, 148, 149

B

BIN - *Binary*. 204

BISMON - . 199–201

BMS - . 111, 112

BRAIN-IoT - *model-Based fRamework for dependable sensing and Actuation in INtelligent decentralized IoT systems*. 208, 215

C

CA - *Certificate Authorities*. 221, 224

CAM - *Compliance Auditing Manager*. 27

CAN - *Controller Area Network*. 54, 55

CAP - *Cybersecurity Assurance Program*. 182

CAPEC - *Common Attack Pattern Enumeration and Classification*. 9, 25, 33

CAS - *Compliance Auditing Service*. 26, 27

CC - *Common Criteria*. 180, 182

CCRA - *Common Criteria Recognition Arrangement*. 182

CCTV - *Closed Circuit Television*. xv, 2

CDPA - *Consumer Data Protection Act*. 18

CERT - *Computer Emergency Response Team*. 9, 37, 175

CERT/CC - *Computer Emergency Response Team/Coordination Center*. 236

CFG - *Control Flow Graph*. 204

CHARIOT - . 196–202, 204, 206, 207, 220–227

CLI - *Command Line Interface*. 223, 225

CMDB - *Configuration Management Database*. 8, 25, 29, 33–35, 39, 42, 43

CNIL - *nationale de l'informatique et des libertés*. 74

CNN - *Convolutional Neural Network*. 52

CNSSI - *Committee on National Security Systems Instruction*. 90

CoAP - *Constrained Application Protocol*. 127

CORAS - . 41, 145

CPA - *Commercial Product Assurance*. 182

CPE - *Common Platform Enumeration*. 9, 25, 33

cPP - *Collaborative Protection Profiles*. 182

CPS - *Cyber-Physical Systems*. 2, 18, 209–212

CPU - *Central Processing Unit*. 75, 211

CRISP-DM - *Cross Industry Standard Process for Data Mining*. 53, 57

CRUD - *Create-Read-Update-Delete (operations)*. 222

CSPN - *Certification de Sécurité de Premier Niveau*. 182, 183

CSS - . 117

CSV - *Comma Separated Value*. 176

CTI - *Cyber Threat Intelligence*. 9, 19, 25, 26

CTIA - *Cellular Telecommunications Industry Association*. 183

CVE - *Common Vulnerabilities and Exposures*. 9, 19, 25, 33, 35

CVRF - *Common Vulnerability Reporting Framework*. 19

CVSS - *Common Vulnerability Scoring System*. 25, 26, 33, 37, 91, 190, 191

CWE - *Common Weakness Enumeration*. 9, 25, 33

CWSS - *Common Weakness Scoring System*. 91

D

DARPA - *Defense Advanced Research Projects Agency*. 53

DBN - *Deep Belief Network*. 52

DCS - *Distributed Control Systems*. 2

DDoS - *Distributed Denial of Service*. xv, 2, 5, 13, 18, 179

DEXi - . 41

DFD - *Data Flow Diagram*. 144, 145, 147–149, 151–153, 156, 157

DID - *Decentralized Identifiers*. 232, 234, 235, 239, 240

DL - *Deep Learning*. 13, 14, 90, 92, 96

DLT - *Distributed Ledger Technology*. xix, 229, 232–235, 238, 243

DNN - *Deep Neural Network*. 97–99

DoS - *Denial of Service*. 5, 38

DPWS - *Devices Profile for Web Services*. 127

DREAD - . 91

DSL - *Domain-specific Language*. 223

DTLS - *Datagram Transport Layer Security*. 94, 213

DVR - *Digital Video Recorder*. xv, 2

E

EAL - *Evaluation Assurance Levels*. 182, 191

EAL4+ - . 183

EAP - *Extensible Authentication Protocol*. 94

EC - *European Commission*. xvi

ECDHE - *Elliptic Curve Diffie-Hellman Ephemeral*. 214

ECSO - *European Cyber Security Organisation*. 180, 191

ECSO WG1 - . 181

ELF - *Executable and Linkable Format*. 204

EMALCSA - . 82

ENISA - . 93, 179, 180

ETAP - . 81

ETL - *Extract Transform Load*. 53

ETSI - *European Telecommunications Standarad Institutes*. 173–175, 180, 184–186, 188, 191, 192

EU - *European Union*. 69, 72, 163, 165, 179, 235

F

FIT IoT-LAB - . 189

G

GCC - *GNU Compiler Collection*. 200

GDPR - *General Data Protection Regulation*. xvi–xviii, 5, 12–14, 18, 26, 69–79, 81, 84, 85, 144, 146, 153, 157, 162–166, 168–173, 175, 177, 183, 235, 247, 248

GHOST - . 163

GNN - *Graph Neural Networks*. 96

GPS - *Global Positioning System*. 38, 54

GUI - *Graphical User Interface*. 27

H

H2020 - *Horizon 2020*. 21, 47, 167

HEX - *Hexadecimal*. 204

HKDF - *HMAC-based Key Derivation Function*. 217

HMAC - *keyed-Hash Message Authentication Code*. 217

HTML - *Hypertext Markup Language*. 117

HTTP - *Hyper Text Transfer protocol*. 227

HTTPS - *Hyper Text Transfer protocol Secure*. 208

HUB - *HUB*. 214

I

IBM - *International Business Machine*. 106, 107, 117

ICASI - *Industry Consortium for Advancement of Security on the Internet*. 19

ICSA - *Institute of Chartered Secretaries and Administrators*. 183

ICT - *Information and Communication Technology*. 73, 89, 90

IDIADA - . 54, 59

IdP - *Identity Provider*. 235

IDS - *Intrusion Detection Systems*. 91, 92

IEC - *International Electrotechnical Commission*. 71–73, 78, 85, 170, 171

IETF - *Internet Engineering Task Force*. 20, 93

IIC - *Industrial Internet Consortium*. 10, 18, 26

IIoT - *Industrial Internet of Things*. 3, 122, 124, 132, 227, 249

IIRA - *Industrial Internet Reference Architecture*. 18

IISF - *Industrial Internet Security Framework*. 10, 18, 50

IODEF - *Incident Object Description Exchange Format*. 20

IODEF-SCI - *IODEF for Structured Cybersecurity Information*. 20

IoT - *Internet of Things*. xv–xx, 1–3, 5, 7–10, 12–14, 17–31, 34, 35, 37–39, 41, 42, 46, 47, 49–53, 58, 65, 66, 69–81, 84, 85, 88–96, 98, 99, 105–115, 117–133, 137, 143, 144, 146, 149, 156, 157, 161–163, 165–167, 170–184, 186, 189, 190, 192, 196–198, 202, 203, 205, 208–218, 220, 221, 225, 226, 229–232, 234–237, 243, 247–249, 251

IoT-A - *Internet of Things Architecture*. 74

IoT-ML - *IoT-Modeling Language*. 74

IoTL - *Internet of Things Language*. 108, 111–113

IP - *Internet Protocol*. 179

IPSE - *IoT Privacy and Safety Engine*. 105, 116–120

IS - *Information Security*. 77

ISO - *International Organization for Standardization*. 71–73, 78, 85, 170, 171, 185

ISRAM - *Information Security Risk Analysis Method*. 145

ISRM - *Information Security Risk Management*. 146

IT - *Information Technology*. 2, 3, 6, 8, 9, 12, 18, 20, 49, 51, 183, 208, 209, 211, 212, 247

J

JSON - *JavaScript Object Notation*. 9, 44, 176, 224

L

LINDDUN - *Linkability, Identifiability, Non-repudiation, Dectectability, Information Disclosure, Content Unawareness, Non-compliance*. xviii, 144, 146–155, 157, 158, 248

LL - . 117

LPWAN - *Low Power Wide Area Network*. 209–211, 213, 214, 218

LSP - *Large Scale Pilot*. xviii

LTE - *Long Term Evolution*. 183

M

M2M - *Machine to Machine*. 218

MAS - *Multi-agent System*. 93, 96, 98

MBT - *Model-based testing approach*. 189

MCU - *Micro Controller Unit*. 213–215

MILE - *Managed Incident Lightweight Exchange*. 20

MITM - *man-in-the-middle*. 175

MITRE - *Massachusetts Institute of Technology Research & Engineering*. 25, 33, 35

ML - *Machine Learning*. xviii, 13, 19, 33, 88, 90, 92, 99

MLP - *Multi-Layer Perception*. 99

MQTT - *MQ Telemetry Transport*. 127

MRA - *Mutual Recognition Agreement*. 182, 183

MUD - *Manufacturer Usage Description*. 92–95, 99

Multi-CAN - *Multi-Controller Area Network*. 38

N

Nadam - *Nesterov-accelerated Adaptive Moment Estimation*. 57

NB-IoT - *NarrowBand-IoT*. 209

NFV - *Network Function Virtualization*. 123, 136

NGIoT - . 177

NIS - *Network Information Systems*. xvi, 26

NIST - *National Institute of Standards and Technology*. 7, 25, 26, 33, 37, 90, 91, 93, 179

NITES - *National IT Evaluation Scheme*. 183

NVD - *National Vulnerability Database*. 26, 91, 187

O

OASIS - *Organization for the Advancement of Structured Information Standards*. 73

OBU - *On Board Unit*. 38

OCL - *Object Constraint Language*. 189

OCTAVE - *Operationally Critical Threat, Asset, and Vulnerability Evaluation*. 145

OECD - *Organization for Economic Cooperation and Development*. 72

OEM - *Original Equipment Manufacturer*. 52

OGC - *Open Geospatial Consortium*. 127

OMA - . 94

OMA LwM2M - *Open Mobile Alliance Lightweight M2M*. 94

OS - . 116, 203

OSCORE - *Object Security for Constrained RESTful Environments*. 94

OT - *Operational Technology*. 2, 3, 18, 209, 210, 247

OVAL - *Open Vulnerability and Assessment Language*. 10

OWASP - *Open Web Application Security Project*. 9, 175

P

PbD - *Privacy-by-design*. 146

PCB - *Printed Circuit Board*. 213

PDF - *Portable Document Format*. 189

PEP - *policy enforcement point*. 134

PIA - *Privacy Impact Assessment*. xvii, 69, 71–75, 79, 81, 84, 85

PII - *Personal Identifiable Information*. 73

PKI - *Public Key Infrastructure*. 111, 113, 119, 213, 217, 220, 221, 223–225

PLC - *Programmable Logic Controllers*. 2, 82

PoW - *Proof of Work*. 12

PP - *Protection Profile*. 182

PSM - *Programming Support Manager*. 27

PSS - *Programming Support Service*. 27, 117

Q

QoS - *Quality of Service*. 89, 90, 96, 123, 130, 133, 134

QR - *Quick Response*. 181, 191

QT - . 54, 61–63

R

RA&M - *Risk Assessment & Mitigation*. 36, 37

RA&MS - *Risk Assessment & Mitigation Service*. 38, 44

RAD - *Risk Assessment Dashboard*. 40

RAE - *Risk Assessment Engine*. 26, 34, 37, 39

RAE DB - *Risk Assessment Engine Dashboard*. 26

ReLU - *rectifier linear unit function*. 56

REST - *Representational State Transfer*. 109, 117

RFID - *Radio Frequency IDentification*. 72

RNN - *Recurrent Neural Network*. 52

ROC - *Receiver Operating Characteristic*. 60, 62, 64

RPC - *Remote Procedure Call*. 224

RPM - *Revolutions per Minute*. 59, 60

RPMA - *Random Phase Multiple Access*. 210

RSD - *Risk Assessment Dashboard*. 26

S

SCADA - *Supervisory Control And Data Acquisition*. 2

SDN - *Software-defined Network*. 92, 93, 99, 122, 131–133, 136

SECaaS - *Security-as-a-Service*. 21, 22, 29, 65

SerIoT - *Secure and Safe Internet of Things*. 88, 89, 95, 99, 100

SFC - *Service Function Chaining*. 135, 136

SIB - *Semantic Information Broker*. 128

SKB - *Security Knowledge Bases*. 8–10, 25, 28, 37

SLA - *Service Level Agreement*. 22, 25, 27–29, 35, 36

SLAAC - *Service Level Agreement Admin Console*. 28

SLAPE - *Service Level Agreement Processing Engine*. 27

SOFIE - *Secure Open Federation for Internet Everywhere*. 244

SOP - *Standard Operative Procedures*. 115

SoTA - *State of The Art*. 71

SPDI - *Security, Privacy, Dependability and Interoperability*. xviii, 121, 123, 128–133, 135–137

SPEP - *Security Policy Enforcement Point*. 23, 24, 26, 37

SPPI - *Security and Privacy Policies Interoperability*. 25

SSN - *Social Security Number*. 127

SSO - *Secure Service Orchestration*. 128

STD - *Security Templates Database*. 24

STIX - *Structured Threat Information Expression*. 19, 20, 33

STRIDE - *Spoofing identity, Tampering, Repudiation, Information disclosure, Denial of service and Elevation of privilege*. xviii, 144, 145, 147, 148, 151–155, 248

T

TITAN - . 189, 190

TLS - *Transport Layer Security*. 94, 213, 214

TMUC - *Trustworthiness Metrics and Utility Calculation*. 25

TOE - *Target of Evaluation*. 182, 185, 186, 188–192

TPM - *Trusted Platform Modules*. 213

TSIS - . 143, 144, 146–148, 153, 156–158

TTCN3 - *Testing and Test Control Notation Version 3*. 189, 190

TTP - *Trusted Third Party*. 10, 11

TXT - *Text File*. 176

U

UAV - *Unmanned Aerial Vehicle*. 50

UID - . 205

UK - *United Kingdom*. 182

UL - *Underwriters Laboratories*. 182

ULD - . 183

UML - *Unified Modeling Language*. 73, 74, 189

UPnP - *Universal Plug and Play*. 127

URL - *Uniform Resource Locator*. 94, 201

V

V2X - *Vehicle-to-everything*. 54

VAE - *Variational Autoencoder*. xvii, 52, 53, 55, 65, 248

VIM - *Virtual Infrastructure Manager*. 136

VNF - *Virtual Network Function*. 135, 136

W

W3C - *World Wide Web Consortium*. 127, 235

WiFi - *Wireless Fidelity*. 72

WP29 - *Working Party article 29*. 164, 165, 171

X

XACML - *eXtensible Access Control Markup Language*. 25

XLS - *Excel Spreadsheet*. 176

XLSX - *Excel Microsoft Office Open XML Format Spreadsheet file*. 176

XML - *eXtensible Markup Language*. 9, 19, 20, 28, 47, 176, 189

XMPP - *eXtensible Messaging and Presence Protocol*. 127

DOI: 10.1561/9781680836837.ch1

Chapter 1

Introduction

By John Soldatos

This chapter is the introduction to the book. It presents the main security and privacy challenges for modern IoT systems, along with state-of-the-art frameworks for security risk assessment and mitigation. Accordingly, it discusses the limitations of these frameworks when it comes to implementing risk management systems for modern IoT systems. Moreover, it outlines some of the enabling technologies and security concepts that could help organizations alleviate these limitations. Some of these concepts are driving the security and technologies that are introduced in subsequent chapters.

1.1 Introduction

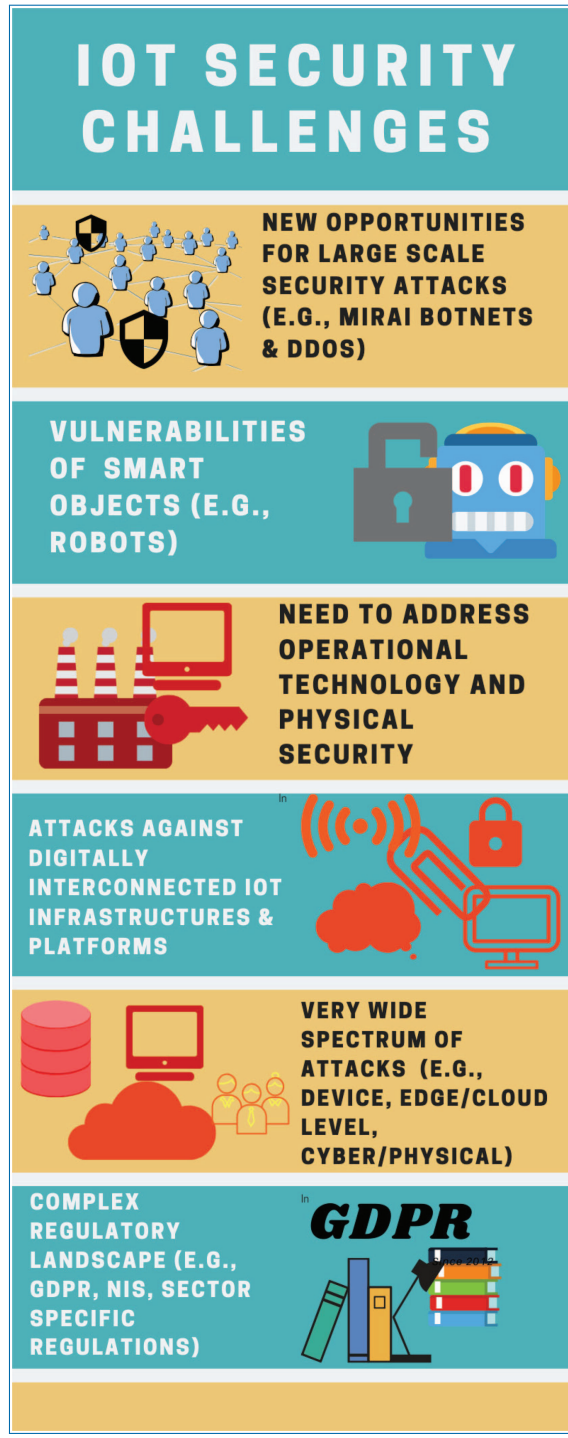
In recent years, we have witnessed the rise of the Internet of Things (IoT) paradigm, which is empowered and propelled by the proliferating number of internet-connected devices. The latter enable a wide range of innovative IoT applications in sectors like trade, transport, healthcare, and industry. A main characteristic of these applications is their ability to take intelligent decisions based on the collection and processing of large amounts of data from the physical world. In several cases, these decisions involve actuation and control activities that influence the status of the physical world and the surrounding environment of the IoT systems.

For over a decade, IoT systems have been evolving in functional and technological sophistication. Early IoT applications were based on the collection and processing of physical world data based on sensors, wireless sensor networks, and other types of internet-connected devices. This early paradigm has gradually evolved in terms of scalability, thanks to the integration of IoT systems and devices with cloud computing infrastructures. The latter infrastructures have also enabled IoT applications to benefit from the elasticity, flexibility, and quality of service of the cloud, which has empowered the development of larger scale applications. Nevertheless, this scalability was based on a heavily centralized model, where very large numbers of internet-connected devices integrate their data and services in the cloud. In recent years, this model has been decentralized based on the emergence and expanded deployment of edge computing architectures and Cyber-Physical Systems (CPS), which include various smart objects like drones, smart wearables, and autonomous guided vehicles. On the one hand, edge computing deployments decentralize IoT functions by placing them closer to the field, while on the other smart objects exhibit (semi) autonomous behaviors that can be implemented independently from the cloud. Hence, emerging IoT applications feature decentralized intelligence, which can be split across the cloud, edge nodes, and smart objects. Edge computing and CPS systems are two of the main pillars of the fourth industrial revolution (Industry 4.0), which closes the loop to the field and enables the digital control of physical processes in a way that blurs the boundaries between the physical and the cyber worlds. Likewise, Industry 4.0 is enabling the convergence of Information Technology (IT) with the Operational Technology (OT) [e.g., Supervisory Control And Data Acquisition (SCADA) systems, DCS (Distributed Control Systems), Programmable Logic Controllers (PLC)] that is widely used in industrial settings.

However, this increased sophistication of IoT systems has introduced a host of security challenges as well, including:

- **New ways for large-scale security attacks:** The proliferation of IoT devices and their deployment has opened new cybersecurity vulnerabilities and enabled new ways of conducting large-scale attacks. As a prominent example, in October 2016, we witnessed the first large-scale distributed denial of service (DDoS) attack based on IoT devices, which took advantage of vulnerabilities (e.g., hard-coded passwords, poorly patched software) of internet-connected CCTV (Closed Circuit Television) cameras and DVR (Digital Video Recorders), in order to deploy the notorious Mirai malware on them [1]. Likewise, back in January 2015, the "Lizard Stressor" attacks compromised many commercial home routers at a large scale.

- **Vulnerabilities in Smart Objects:** As "things" get more sophisticated, automated, and interconnected, their cyber resilience becomes more critical than ever before. This is important as adversaries can nowadays create significant damage by attacking individual smart objects and internet-connected devices.
- **Threats associated with OT (Operational Technology) and Physical Security:** The convergence of IT and OT in industrial environments asks for new integrated approaches to security that protect cyber and physical assets at the same time. Indeed, OT systems are characterized by poor cybersecurity, as OT protocols are usually vendor specific and not designed for security. Moreover, OT environments are sometimes supported by old and insecure IT systems (e.g., old operating systems, vulnerable drivers), while providing very poor documentation regarding their security-related characteristics. Nowadays, many malicious parties try to compromise physical assets (e.g., a data center) in order to attack cyber assets. In some cases, they also attempt the so-called combined attacks, which go against cyber and physical assets at the same time.
- **Attack opportunities against digitally interconnected IoT infrastructures:** IoT infrastructures in various sectors are gradually becoming digitally interconnected, as part of value chains in different sectors. For example, different Industrial IoT (IIoT) assets residing in factories and logistics infrastructures are getting interconnected as part of the manufacturing value chain. As another example, IoT-enabled healthcare platforms are becoming digitally interconnected with intelligent transport platforms to enable the delivery of emergency services. The interconnection of IoT platforms and infrastructures in different sectors increases their dependencies and provides the means for attacking an infrastructure by launching an attack to an interconnected infrastructure. Once one infrastructure is attacked, its interconnected counterparts suffer from cascading effects. To alleviate such effects, coordination and orchestration of diverse, decentralized cybersecurity platforms and methodologies is required. This goes typically beyond the capabilities of state-of-the-art platforms for IoT security.
- **A very wide spectrum of attacks:** Non-trivial IoT systems comprise complex IT infrastructures including networking, computing systems, software/middleware systems, cloud computing systems, as well as a host of internet-connected devices. Therefore, cybersecurity solutions have to protect a very wide range of assets against a host of different attacks such as scanning and mapping attacks against wireless and/or wired networks, protocols attacks, data theft and loss of confidentiality attacks, attacks against


IOT SECURITY CHALLENGES
NEW OPPORTUNITIES FOR LARGE SCALE SECURITY ATTACKS (E.G., MIRAI BOTNETS & DDOS)
VULNERABILITIES OF SMART OBJECTS (E.G., ROBOTS)
NEED TO ADDRESS OPERATIONAL TECHNOLOGY AND PHYSICAL SECURITY
ATTACKS AGAINST DIGITALLY INTERCONNECTED IOT INFRASTRUCTURES & PLATFORMS
VERY WIDE SPECTRUM OF ATTACKS (E.G., DEVICE, EDGE/CLOUD LEVEL, CYBER/PHYSICAL)
COMPLEX REGULATORY LANDSCAPE (E.G., GDPR, NIS, SECTOR SPECIFIC REGULATIONS)
GDPR

cryptographic algorithms and key management systems, spoofing and other authentication attacks, attacks against operating systems and applications, Denial of Service (DoS) and jamming attacks (including DDoS), access control attacks, as well as physical security attacks.

- **A demanding regulatory landscape:** Integrators of IoT solutions and providers of IoT services must nowadays comply with stringent directives and regulations, which they cannot afford to ignore. As a prominent example, in Europe the General Data Protection Regulation (GDPR) imposes strict requirements for personal data protection, along with extremely high penalties for cases of non-compliance. GDPR has a global impact, as several countries are already using GDPR as a blueprint for updating their data protection laws and regulations.

Recent large-scale cybersecurity attacks against IoT infrastructures have demonstrated that these challenges are for real. Moreover, they have illustrated that organizations that adopt and deploy IoT technologies are exposed to significant operational risk that can have severe economic impact.

1.2 Overview and Limitations of Security Risk Assessment Frameworks

1.2.1 Overview of Security Risk Assessment

A security risk assessment process is one of the most important activities that help organizations identify and mitigate the operational and the economic impact of security threats. Specifically, the process of security risk assessment identifies, assesses, and implements security controls in applications. One of its main goals is to anticipate and prevent security vulnerabilities and threats. When carrying out a security risk assessment, organizations can examine their assets end to end, while considering the attacker's perspective. The outcome of the risk assessment process is typically a comprehensive report that can support the business management of the organizations in their decision-making. Specifically, by considering the risks that are associated with the various assets and the likelihood of their occurrence, managers can make informed decisions about resource allocation, procurements of security solutions, as well as about security controls implementations.

A typical security risk assessment process entails the following steps:

- **Identifying the assets that are most valuable to the company.** In the case of IoT security risk assessment, the assets include IoT devices and IoT services.

- **Determining the potential threats to each asset.** This step leverages information about the IoT asset (e.g., information collected from a device via an appropriate probe).
- **Estimating the likelihood that events associated with the threat will occur.** Usually, a probability estimate is produced based on some deterministic or stochastic model about the asset and the attacker's behavior.
- **Determining the impact criticality of each loss event.** Events that could have a catastrophic impact on the assets and the organization should be avoided as a matter of priority.
- **Calculate risk scores** based on combinations of information about events' probabilities and their impact criticality.

In-line with the above-listed steps, risk management methods rely on the estimation of a risk situation for the various assets. This estimation is based on information about the processes, the assets, and the infrastructure of an organization. The main goal of the process is the identification of potential risks and the development of appropriate protective measures. The tangible outcome of a risk analysis is in most cases a list of risks or threats to a system, together with the corresponding probabilities.

International standards in the field of risk management are used to support the identification of risks or threats, and to assess their respective probabilities. These standards range from general considerations and guidelines for risk management processes (e.g., [2, 3]) to specific guidelines for the IT sector such as the ISO20000 [4], ISO27000 [5], ISO/IEC 27002 [6], ISO27005 [7], and others (e.g., [8]), all the way to highly specific frameworks. Most of these standards specify framework conditions for the risk management process, but rarely go into detail on specific methods for the risk analysis or risk assessment. This is one reason why often differences in the risk assessment arise within the specific areas of an application, making a direct comparison of the results of the various frameworks difficult.

In this context, choosing the right method and the right tool for risk analysis and risk evaluation proves to be complicated. In recent years, various concepts, algorithms, and tools have evolved from research, specially designed to protect IT infrastructures and systems. Since their historical background is settled in a business context, in these methods a quantitative risk assessment is usually performed based on monetary costs [9, 10]. This is the case with the French EBIOS[1] and the UK's

1. EBIOS stands for Expression des Besoins et Identification des Objectifs de Sécurité – Expression of Needs and Identification of Security Objectives.

CRAMM[2] methods, but also with the ISO/IEC 27005:2013 standard [7]. Most of the risk assessment methods and tools (e.g., most of the ones listed in [11]) use the commonly known rule of thumb "risk = probability × potential damage" [12]. Depending on the applied method, the terms and scales for the assessment of the probabilities as well as the potential damage are predefined (such as in the NIST (National Institute of Standards and Technology) policy [13] or in the Mehari method [14]). In practice, the selection of a specific risk-assessment tool is based on practical considerations and depends on how well the present terminology of the application can be mapped onto the predefined specific terminology of the risk assessment methodology. In order to structure the process of risk assessment, there are various attempts to develop models for general risk assessments. For example, the AURUM system [15] provides a graphical tool for the risk modeling based on ontologies. It uses a Bayesian approach for determining threat probabilities (which is also done by the method proposed in [16]). The OCTAVE method [17] is based on subjectively estimated probabilities and thus can be understood as an a priori distribution with regards to the Bayesian approach. Also, the CORAS method [18] allows the integration of several different risk assessment processes, whereas the identification of the probability of an attack is not done automatically but a priori to any risk assessment.

Finally, the assessment per se is performed in order to optimize the existing configuration of an operation environment. A variety of mathematical models that embed different reasoning techniques have been proposed, such as Integer Linear Programming, Constraint Programming, Stochastic Optimization, Multi-criteria Optimization, Robust Optimization, Metaheuristics, etc. In the literature, a variety of optimization and decision support approaches have been reported. They integrate techniques and models from Artificial Intelligence, Operations Research, and Computational Logics (e.g., [19–21]).

1.2.2 Limitations of Security Risk Assessment Frameworks for IoT

For nearly two decades, various organizations have used the above-listed methods and frameworks for cybersecurity risk assessment. In general, these frameworks have been proven efficient in protecting organizations from security risks. However, they fall short when it comes to protecting modern IoT systems, which introduce a whole new range of assets such as IoT devices and a cyber-physical (rather than a

2. CRAMM (CCTA Risk Analysis and Management Method) is a qualitative risk analysis and management tool developed by UK government's Central Computer and Telecommunications Agency (OGC since April 2001).

cyber only) nature. Specifically, IoT systems present unique challenges that cannot be adequately tackled based on conventional approaches. These include:

- **Increased Dynamism, Variability, and Scale:** Non-trivial IoT deployments comprise very large number of devices, which are in many cases very densely connected. These characteristics make them very different from conventional IT systems and drive legacy risk assessment approaches to their limits. For example, the dynamic nature of IoT environments imposes a need for continuous assessment of the state of the IoT assets rather than the commonly used periodic assessments.
- **Inter-enterprise Connectivity across IoT systems:** IoT systems are increasingly becoming interconnected. This is, for example, the case with Industrial IoT platforms across the value chains of industrial sectors (e.g., supply chains in smart manufacturing). It's therefore common for the risk status of an asset to be determined based on the status of other interconnected assets. As a result, security risk assessments need to consider the interconnected nature of IoT assets, including possibilities for collaborative risk assessments and the impact of cascading effects.
- **Cyber-Physical Nature of the IoT Infrastructures:** Complex IoT infrastructures can be considered as large-scale cyber-physical systems. They comprise both cyber and physical assets, which provide the means for interacting with the field and for influencing the status of the physical world. In this context, risk assessments should consider the cyber-physical nature of IoT systems, including both cyber and physical risks. Likewise, risk assessments should consider combined cyber-physical attacks (e.g., launching an attack against a physical asset in order to compromise a physical asset subsequently). Managing cyber-physical risks is challenging as a result of the complexity and interdependencies among various system components, including the integration between communication, computing and control technology.
- **Need for IoT Cybersecurity Knowledge:** In order to assess the risk of a system, cybersecurity officers need to have some baseline knowledge about the IoT assets. In practice, such knowledge is accumulated within a Security Knowledge Base (SKB) or a Configuration Management Database (CMDB). In the case of IoT systems, the knowledge to be attained is much more sophisticated than the corresponding knowledge about conventional IoT systems. For example, knowledge about the vulnerabilities of IoT devices is required, as well as about how threats and vulnerabilities on one IoT asset affect another. This complicates the security risk assessment task and renders some of the conventional methods and knowledge bases inadequate (e.g., prone to errors and likely to miss some important risk).

- **Complexity and Scale as Drivers for Increased Automation:** The rising complexity of IoT environment asks for increased automation in the risk assessment process. Hence, conventional manual assessments should be replaced by software systems like machine learning and artificial intelligence.
- **New Organizational and Social Elements:** Much as risk assessments are about technical measures and risks, there are also about establishing proper structures and processes. IoT environments ask for novel organizational structures and processes, beyond conventional ways for organizing security functions and teams (e.g., CERTs (Computer Emergency Response Teams)).

1.3 New Technology Enablers and Novel Security Concepts

To address the above-listed limitations, cybersecurity experts and system integrators are nowadays offered with novel technologies that enable effective solutions for IoT security risk assessments. Following paragraphs outline some of these technologies and the problems that they can solve.

1.3.1 IoT Security Knowledge Bases

Security Knowledge Bases (SKBs) are very important and versatile tools for matching identified threats and incidents to existing knowledge during the execution of security services. SKBs consolidate several sources of knowledge for Cyber Threat Intelligence (CTI), such as:

- **CPE (Common Platform Enumeration),** which is a structured naming scheme for IT software, systems, and packages.
- **CWE (Common Weakness Enumeration),** which lists common software's vulnerabilities.
- **CAPEC (Common Attack Pattern Enumeration and Classification),** which lists common attack patterns on software and their taxonomy.
- **CVE (Common Vulnerabilities and Exposures),** which lists all publicly known cybersecurity vulnerabilities and exposures.

SKBs can also collect and store external CTI data sources through available documents in various formats like JSON (JavaScript Object Notation) and XML (eXtensible Markup Language). There are several SKBs, including commercial SKBs from major security vendors and SBKs from standards development bodies (e.g., the OWASP (Open Web Application Security Project) Security Knowledge Framework). Nevertheless, these SKBs are general purpose and do not comprise

suitable security knowledge databases for the IoT assets, which is a setback to supporting security applications like risk analysis and security management activities for IoT environments.

In order to enable effective and automated risk assessments for IoT systems, SKBs should also collect and consolidate information from additional sources, as means of covering IoT assets. For example, the OVAL (Open Vulnerability and Assessment Language) specifications enable the description of IoT assets.

1.3.2 IoT Reference Architectures and Security Frameworks

To extend existing security risk assessment systems to the IoT space, organizations can take advantage of readily available reference models and architectures for IoT systems. One such architecture is the Reference Architecture of the Industrial Internet Consortium (IIC), which is accompanied by a security framework, namely the Industrial Internet Security Framework (IISF) [22]. These reference models provide insights for implementing general purpose security measures for large-scale IoT systems. Furthermore, they provide ideas and guidelines about implementing security measures across entire supply chains that span multiple IoT platforms. In an era where infrastructures are becoming digitally interconnected, an attack on one part of the supply chain can have severe effects on others. While the issue of supply chain security is known and relevant standards exist (e.g., ISO 28000 [23]), practical implementations are still in their infancy. To address the issue of supply chain security, there is a need for seamless information sharing and collaboration between supply chain systems and stakeholders.

Following chapters of the book present end-to-end security solutions that adhere to the principles of the above-listed IoT reference architectures and security frameworks.

1.3.3 Blockchain Technology for Decentralized Secure Data Sharing for Security in IoT Value Chains

To secure IoT-based supply chains, it is essential to facilitate information sharing between supply chain participants, as means of implementing security functionalities such as collaborative risk assessment. The most prominent way for sharing information between the various stakeholders involves the use of a shared database, where selected information from all the different supply chain participants will be persisted and accessed. Nevertheless, centralized information sharing is associated with prominent limitations including:

- **The need for a Trusted Third Party (TTP)** that owns and operates the centralized database infrastructure. However, in IoT supply chains, there is not always a prominent and accepted TTP.

- **Persisting shared information in a centralized infrastructure makes it susceptible to cybersecurity attacks.** Indeed, a centralized database once compromised would infect and destabilize the information sharing system.

In recent years, decentralized paradigms to sharing information have emerged, including blockchain-based approaches. Blockchains present some unique opportunities for securely sharing information across parties such as [24]:

- **A distributed architecture that can deter or minimize the effect of cyberattacks against the shared information.** A distributed network structure provides inherent operational resilience as there is no single point of failure.
- **A consensus validation mechanism for news blocks of (shared) data,** which enables a continuous check on the integrity of past transactions and guarantees strong security and decentralized trust. In the scope of a blockchain mechanism, there is no need for a TTP to authenticate and validate ownership of the data, as the nodes in the overall network use the peer-to-peer network to process and verify transactions.
- **Strong encryption features,** as blockchain networks employ multiple forms of encryption at different points and provide multi-layered protections against cybersecurity threats. Participants access rights are secured through asymmetric-key cryptography or public/private key encryption.
- **Transparency,** which provides another degree of cybersecurity protection, as it is more challenging for hackers to place malware in the network to collect information and to transmit it covertly to another database managed by the hacker [25]. Each participant has an identical copy of the ledger, and hence, the network creates the opportunity for deploying enhanced compliance processes including, among other things, real-time auditing or monitoring. As a result, vulnerabilities and threats may be identified quickly if good risk management and compliance controls are implemented.
- **Opportunities for securing connected devices and sharing information from them,** through making them peer nodes in the blockchain network [26].
- **Support for Smart Contracts,** which can implement security logic (i.e., security workflows) over the shared information.

In practice, these benefits can be better leveraged in the scope of permissioned blockchain networks rather than in the scope of the public blockchain infrastructures that support blockbuster cryptocurrencies like Bitcoin and Ethereum. The main reason for this is that permissioned blockchains offer authentication and privacy control of the participants, along with much better performance (i.e., reduced

latency and handling of more transactions per second). The latter benefits stem from the fact that permissioned blockchains can be decoupled from the computationally expensive Proof of Work (PoW) mechanisms. Moreover, permissioned blockchains are often hosted on cloud platforms that have robust cybersecurity controls across different layers of the technology stack.

The use of permissioned blockchain networks for sharing security information is not widespread, and security implementations are in early stages. For example, the authors in [27] describe an early implementation of blockchain-based solution for sharing security information across financial institutions. There are also blockchain solutions for information sharing in general, as well as for protecting specific IT systems and IoT devices (e.g., [26]) and enabling the trusted exchange of data between them (e.g., [28]).

Later chapters of the book present practical uses of blockchain technology for privacy and data protection in IoT systems.

1.3.4 Technologies Facilitating GDPR Compliance

GDPR mandates the principle of proportionality (also called purpose limitation). According to this principle, data should only be processed for the purpose for which it was collected and from which consent has been provided. Related technology focuses on various consent management methods, authentication, and data anonymization. Modern privacy-preserving encryption methods can potentially give mathematical guarantees for such privacy and thus provide more rigorous protection. Recently, there have been significant advancements in privacy-preserving encryption methods, which are gradually moving from research implementations to enterprise deployments. For example, functional encryption solutions enable the implementation of IoT services that cannot read the encrypted data, yet they are able to read the results of the computation of specific functions of the data. In this way, functional encryption can automatically safeguard GDPR requirements, such as the need to use private data only for the purpose for which it has been provided. As another example in the area of encryption, homomorphic encryption solutions enable a service to perform computations on encrypted private data and then return the encrypted result to the user who can decrypt it to read the results.

There are many more technologies and security approaches that can facilitate GDPR compliance. For example, multi-factor authentication techniques are commonly deployed along with encryption to prevent data breaches and boost GDPR compliance. As another example, there are also pseudonymization technologies that help masking personal data and avoiding the pitfalls of non-compliance. Pseudonymization is referenced in the text of the GDPR itself. There are also other

technologies that facilitate the tracking and mapping of data workflows in order to identify incidents like data loss, data theft, and security attacks.

Several chapters of the book illustrate GDPR compliant solutions, along with methodologies (e.g., privacy by design) for implementing and deploying them in practical IoT security contexts.

1.3.5 Machine Learning and Artificial Intelligence Technologies for Data-driven Security

For nearly two decades, several research works made use of Machine Learning (ML) and Artificial Intelligence (AI) to analyze large amounts of security data as a means of deriving insights on potential threats, vulnerabilities, and risks. IoT devices and infrastructures generate large volumes of data and hence could be analyzed based on similar approaches. This requires the application of ML algorithms over IoT security datasets. In this context, the use of Deep Learning (DL) for detecting anomalies and potential intrusions is suggested given that the performance of DL algorithms improves significantly when trained with very large datasets. The research literature includes many works that apply ML/DL techniques towards detecting security issues and anomalies at the network level, i.e., techniques applied over network layer datasets. Nevertheless, such data mining techniques have not been extensively applied in the case of IoT applications. The growing popularity of ML, DL, and AI provides opportunities for building data-driven systems that will detect security issues timely and effectively.

Several chapters of the book present data-driven approaches to IoT security. One chapter presents the more specific case of applying DL for detecting anomalies in IoT-related datasets.

1.4 Conclusion

This chapter has presented some of the most prominent cybersecurity challenges for modern IoT environments. It has also illustrated how some of the main characteristics of IoT systems are driving these challenges. Furthermore, it has presented an overview of security risk assessment functionalities, including the standards and frameworks that support their implementation. State-of-the-art techniques for security risk assessment have prominent limitations when it comes to confronting risks associated with large-scale, cyber-physical, and interconnected IoT systems. Recent attacks against IoT systems (e.g., the Mirai-based DDoS attack) have manifested these limitations and opened the discussion for new approaches. In this direction, security systems integrators can benefit from a range of novel technologies

that can boost the timeliness and efficiency of the risk assessment processes, while dealing with other concerns like GDPR compliance. For example, ML/DL technologies can detect complex attack patterns in order to accelerate the preparedness of security teams. Likewise, blockchain techniques can boost the transparency of data sharing across connected IoT systems. Furthermore, IoT security knowledge bases can increase the automation of the IoT risk assessment process, by helping organizations to identify IoT-specific threats and to provide advanced cyber threat intelligence for IoT systems. Following chapters of the book illustrate a host of novel techniques for IoT security, which leverage advanced technology enablers such the ones outlined above.

Acknowledgments

Part of this work has been carried out in the scope of the H2020 SecureIoT project, which is funded by the European Commission in the scope of its H2020 program (contract number 779899).

References

[1] John Soldatos, "Why the DDoS attack happened and what can be done to prevent more episodes", The Internet of All Things. October, 2016 available at: https://www.theinternetofallthings.com/why-the-ddos-attack-happened-10262016/ [Accessed: February 2020].
[2] International Standardization Organization, "ISO 31000: Risk Management – Principles and Guidelines", Geneva, Switzerland, 2009.
[3] International Standardization Organization, "ISO 31010: Risk management – Risk assessment techniques", Geneva, Switzerland, 2009.
[4] International Standardization Organization, "ISO 20000: Information Technology Service Management", Geneva, Switzerland, 2005.
[5] International Standardization Organization, "ISO 27001: Information Security Management System Requirements", Geneva, Switzerland, 2013.
[6] International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC), "ISO/IEC 27002 Information technology – Security techniques – Code of practice for information security management", 2005.
[7] International Standardization Organization, "ISO 27005: Information security risk management", Geneva, Switzerland, 2011.

[8] Common Criteria Working Group, "Common Methodology for Information Technology Security Evaluation – Evaluation methodology", CCMB-2007-09-004, http://www.commoncriteriaportal.org, 2007.
[9] T. R. Peltier, "Information security risk analysis", Auerbach Publications, 2001.
[10] S. E. Schechter, "Computer security strength and risk: a quantitative approach", Harvard University, 2004.
[11] European Network and Information Security Agency, "Inventory of Risk Management/Risk Assessment Methods", 2010. [Online], Available: rm-inv.enisa.europa.eu/rm_ra_methods.html
[12] CCRA Working Group, "Common Criteria for Information Technology Security Evaluation", CCRA, available at: www.commoncriteriaportal.org, 2006.
[13] G. Stoneburner, A. Goguen and A. Feringa, "Special Publication 800-30: Risk Management Guide for Information Technology Systems", National Institute of Standards and Technology, 2002.
[14] Clusif Methods Commission, "MEHARI V3 Risk Analysis Guide", 2004.
[15] A. Ekelhart, S. Fenz and T. Neubauer, "Automated Risk and Utility Management", in Proceedings of the Sixth International Conference on Information Technology: New Generations, IEEE Computer Society, 2009, pp. 393–398.
[16] F. Foroughi, "Information Security Risk Assessment by Using Bayesian Learning Technique", in Proceedings of the World Congress on Engineering, Bd. 1, International Association of Engineers, 2008, pp. 2–6.
[17] C. J. Alberts and A. Dorofee, "Managing Information Security Risks: The Octave Approach", Addison-Wesley Longman Publishing Co., Inc., 2002.
[18] EU Project Nr. IST-2000-25031, "CORAS - Risk Assessment of Security Critical Systems", Online. Available: http://www2.nr.no/coras/, 2003.
[19] M. Milano (ed.). Constraint and Integer Programming: toward a unifying methodology, Kluwer Ac. Publisher, 2004.
[20] J. N. Hooker: A Hybrid Method for the Planning and Scheduling. Constraints, 10(4): 385–401 (2005).
[21] L.-M. Rousseau, M. Gendreau, G. Pesant and F. Focacci: Solving VRPTWs with Constraint Programming Based Column Generation. Annals OR, 130(1–4): 199–216 (2004).
[22] The Industrial Internet Security Framework Technical Report, available at: https://www.iiconsortium.org/IISF.htm (Accessed February 2020).
[23] ISO 28000:2007 Specification for security management systems for the supply chain, available at: https://www.iso.org/standard/44641.html (Accessed February 2020).

[24] N. Kshetri, "Blockchains roles in strengthening cybersecurity and protecting privacy", Telecomm. Policy, 41(10): 10271038 (2017).
[25] J. Gu, B. Sun, X. Du, J. Wang, Y. Zhuang and Z. Wang, "Consortium blockchain-based malware detection in mobile devices", IEEE Access, 6 (2018), p. 1211812128.
[26] O. Alphand, *et al.* "IoTChain: A Blockchain Security Architecture for the Internet of Things", IEEE Wireless Communications and Networking Conference (WCNC) (2018), pp. 1–6.
[27] I. Karagiannis, K. Mavrogiannis, J. Soldatos, D. Drakoulis, E. Troiano and A. Polyviou: Blockchain Based Sharing of Security Information for Critical Infrastructures of the Finance Sector. IOSec/MSTEC/FINSEC@ESORICS 2019: 226–241.
[28] Z. Huang, X. Su, Y. Zhang, C. Shi, H. Zhang and L. Xie, "A decentralized solution for IoT data trusted exchange based-on blockchain", 2017 3rd IEEE Int. Conf. Comput. Commun. (2017), p. 11801184.

DOI: 10.1561/9781680836837.ch2

Chapter 2

Security Data Modelling for Configurable Risk Assessment as a Service in IoT Systems

By Nikos Kefalakis, Angela-Maria Despotopoulou, Spyridon Evangelatos and John Soldatos

The implementation of data-driven security mechanisms for IoT systems requires the collection and analysis of large amounts of security-related data from various heterogeneous sources and Probes. This requires the unification and consolidation of the diverse semantics of these data, based on the use of a common format for representing cyber threat intelligence. However, state-of-the-art formats for modeling and exchanging cybersecurity data are heavy weight and not tailored to IoT systems. Moreover, they do not provide the means for modeling and configuring the IoT security data collection system. This chapter introduces a data model for IoT security data collection systems, which enables the implementation of data-driven security services for non-trivial IoT systems, such as risk assessment services. The introduced data model provides also the means for configuring data-driven IoT security platforms, including their Probes and data collection services. In addition to presenting the main entities of the model and the relationships between them, the chapter illustrates the use of the model for the implementation of a risk assessment as a service platform. Overall, the IoT security data model of the chapter can serve as a basis for implementing lightweight and configurable data-driven solutions for non-trivial IoT systems.

2.1 Introduction

In recent years, the rapid proliferation of internet-connected devices is driving an unprecedented growth of Internet of Things (IoT) systems and applications. The latter are evolving both in terms of the functionalities that they provide to end users, but also in terms of the technologies that they comprise. Specifically, non-trivial state-of-the-art IoT systems comprise numerous devices, including not only conventional sensors and passive devices, but also smart objects and cyber-physical systems (CPS) like drones, automated guided vehicles and industrial robots. Moreover, IoT systems are deployed in scalable edge/cloud computing infrastructures, which comprise broadband networks, edge gateways and devices, as well as cloud data centers. The rising sophistication of IoT systems provides the means for developing and deploying novel IoT applications, yet it also introduces significant cybersecurity challenges [1]. These challenges are evident in the scope of recent cybersecurity incidents against IoT infrastructures and services. As outlined in the introductory chapter of the book, these challenges include:

- The emergence of new types of large-scale security attacks against IoT systems, such as the large-scale Distributed Denial of Service (DDoS) attacks that exploit the vulnerabilities of specific IoT devices.
- Vulnerabilities in cyber-physical systems (CPS), which are associated with the cyber resilience challenges of CPS systems.
- Security threats associated with the interplay between Information Technology (IT) and Operational Technology (OT) infrastructures, given the different nature of IT and OT assets and the conflicting IT and OT security requirements.
- Complex regulatory compliance requirements in a demanding and volatile landscape, such as the need for compliance to the General Data Protection Regulation (GDPR) in Europe and the Consumer Data Protection Act (CDPA) in California, in the U.S.

In order to cope with these challenges, there is a need for end-to-end solutions and tools that can protect IoT assets from the many different types of attacks. Security monitoring and security data analytics is one of the primary functionalities in this direction. It is also one of the security building blocks of standards-based IoT architectures, such as the Industrial Internet Reference Architecture (IIRA) of the Industrial Internet Consortium (IIC). The IIRA has been specified along with an accompanying security framework, namely the Industrial Internet Security Framework (IISF) [2]. The IISF specifies monitoring, analysis, and action workflows as a core security functionality for IoT systems. Based on such IoT monitoring

workflows, it is possible to monitor the various devices, systems, and services that comprise an IoT system in effective and integrated ways. Accordingly, monitoring can then be a foundation for the identification, assessment, and mitigation of risks, i.e., for the implementation of risk management functionalities that are an integral and important element of every non-trivial security policy.

The implementation of data-driven security mechanisms is based on the employment of advanced analytics of large amounts of security data for the various assets that comprise the IoT system [3]. These analytics include the application of machine learning (ML) and artificial intelligence (AI) techniques, which can enable the extraction of insights about security incidents, vulnerabilities, and risks, as a means of executing risk assessment functions and boosting the early and proper preparedness of security teams [4]. Likewise, ML and AI algorithms can be used for extracting IoT security knowledge, including insights not already known. As a prominent example, the use of predictive analytics can enable the extraction of predictive security insights about IoT assets, in order to facilitate timely preparedness.

The collection and consolidation of security data from multiple heterogenous sources is a key prerequisite for the design and implementation of data-driven security techniques, including ML/AI algorithms for security analysis [5]. Indeed, security data stem from a variety of different Probes that are deployed in the diverse assets and devices of an IoT system. These Probes provide data in different semantics and formats, which makes it difficult to consolidate and analyze them in a common and unified way. This is a setback to the implementation of integrated, intelligent, data-driven security techniques, which are capable of correlating information and events associated with different assets, as a means of identifying vulnerabilities and attack patterns beyond existing security knowledge [e.g., knowledge available in popular vulnerability databases like Common Vulnerability and Exposures (CVE)] [6].

In order to remedy this situation, there is a need for collecting and mapping security data in a common model for IoT security knowledge modeling and cyber threat intelligence. Such a common data model can be a foundation for security interoperability of IoT security data monitoring systems that collect and analyze information from different platforms and devices [5, 7]. In this direction, standards organizations have developed knowledge models for cyber threat intelligence like STIX (Structured Threat Information Expression) [8], which serves both as a language and as a serialization format that enables the exchange of cyber threat intelligence (CTI). As another example, the Industry Consortium for Advancement of Security on the Internet (ICASI) has specified the Common Vulnerability Reporting Framework (CVRF) [9]. CVRF is an XML-based data exchange format that automates data reporting and consumption of information about software vulnerabilities, while providing support for describing the vulnerability handling

life cycle end-to-end. Likewise, the Managed Incident Lightweight Exchange (MILE) working group of the Internet Engineering Task Force has introduced the Incident Object Description Exchange Format (IODEF), which is described in RFC5070 [10]. It is an XML format for exchanging operational and statistical security incident information. It can report information about hosts, networks, services, attacks methodologies and forensic data. As part of RFC7203 [11], a set of extensions over IODEF have been specified, namely the IODEF-SCI (IODEF for Structured Cybersecurity Information) format. IODEF-SCI embeds structured information within IODEF and defines new classes for types of information that are not included in IODEF. The above list of formats for exchange, reporting, and serialization of cyber threat intelligence information is non-exhaustive. There are various other standards-based formats, which focus on more specific types of security incidents (e.g., abuse of messages) and in specific layers of the cyber infrastructure (e.g., networking layer). However, most of these formats tend to be heavyweight and their use in IoT security solutions is typically associated with a considerable learning curve for IoT vendors and solution providers. Furthermore, most of them are not tailored to IoT systems and devices, but rather cover a broader range of IT systems and devices. Moreover, these formats are completely agnostic to the data collection system, i.e., the Probes that are used to collect the information. Hence, they do not provide the means for (re)configuring the operation of the IoT security system in order to implement security automation functions. For example, formats like STIX do not provide the means for defining and executing actionable intelligence on the IoT security system, e.g., towards increasing the amount of collecting data or invoking actuating functions in response to the identification or assessment of certain risks. This could greatly facilitate the implementation of security automation functions, through combining reporting of security threats with actionable intelligence on the security infrastructure.

Given these limitations of existing standards and formats, this chapter introduces a new lightweight model for modeling assets and security functions in IoT systems. This model is already used to provide security functionalities for IoT systems in various use cases in the areas of connected transport, healthcare, and smart manufacturing. Due to its lightweight nature, the model sacrifices expressiveness and rich semantics, in exchange of simplicity and a smoother learning curve for IoT developers and solution integrators. Furthermore, it is tailored to IoT systems and services, rather than prioritizing the modeling of other types of cyber assets and cyber infrastructures. Also, it has the unique property of blending information about the actual security data collection systems (e.g., the IoT security Probes and their configurations) with threat intelligence information. This property facilitates the implementation of actionable intelligence, through enabling the encoding of security actions in response to identification of security incidents and vulnerabilities, or even in response to security risk assessment functions. The

security model that is presented in the chapter has been designed, developed, and validated in the scope of H2020 SecureIoT project [5], which provides end-to-end predictive IoT security solutions for various use cases, along with security services like risk assessment and compliance auditing. As already outlined, the presented data model is used for modeling security data for various use cases in connected transport, healthcare, and smart manufacturing. At the same time, it is used for configuring the SecureIoT data collection platform, as a means of facilitating different deployment configurations at both design and run time. These configurations boost the implementation of actionable security intelligence, as well as allow the security services of the platform to reconfigure the operation of the various Probes. This can also enable scenarios that close the loop to the field upon the identification or the grading of certain security risks.

The rest of the chapter is structured as follows: Section 2.2 introduces the architecture of the SecureIoT platform [5], which has driven the specification of some of the entities of the data model. Section 2.3 presents the data model, including its main entities and the relationships between them. The presentation focuses on the structural characteristics of the model and its constructs that enable data collection and security data analysis. Note, however, that an exhaustive presentation of the data model, including details for each one of the entities and their attributes, is beyond the scope of this chapter. Section 2.4 is devoted to the description of the SecureIoT Risk Assessment infrastructure, with emphasis on how it leverages the introduced data model. This infrastructure is currently used to support multiple IoT security use cases based on different instantiations of the data model. Section 2.5 is the final and concluding section of the chapter.

2.2 Data-driven Security Architecture

2.2.1 Overview

The SecureIoT project developed a BigData architecture for the security monitoring and protection of IoT systems at scale. The architecture serves as a blueprint for the development and deployment of IoT security systems. It enables the offering of security services (e.g., risk assessment, compliance auditing) based on a managed security, cloud-based model, i.e., a SECaaS (Security-as-a-Service) model.

The SecureIoT architecture specifies various components, which are grouped together based on their functional relevance. As shown in Figure 2.1, the components' groups of the SecureIoT architecture are:

- **Data Management Group:** This group comprises components that collect security-related data from IoT systems, as well as components that implement actuation and security automation functions.

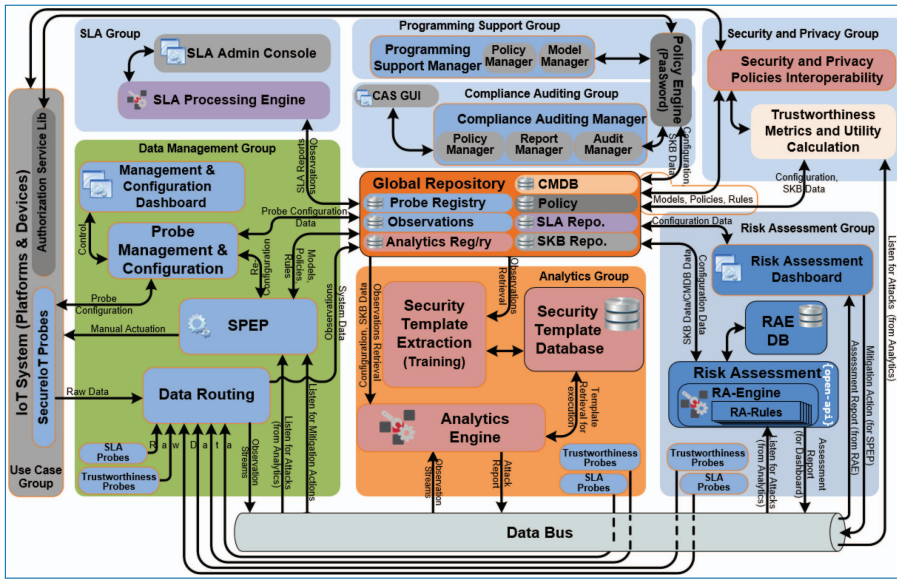


Figure 2.1. SecureIoT architecture for data-driven security services for IoT systems [5].

- **Analytics Group:** This group clusters components that process and analyze security data in order to derive insights that are accordingly used for security risk assessments, compliance auditing and other security functionalities. Data analytics are supported by a powerful mechanism of reusable "security templates," which addresses different types of threats, vulnerabilities, and attacks.
- **Global Repository:** This group comprises databases and repositories where information generated from the various components is persisted. The global repository is at the center of the architecture and interfaces to all the different/other groups of components.
- **SLA Group:** This group comprises components that keep track and audit Service Level Agreements (SLAs) between operators of IoT systems and the SecureIoT services providers in the scope of the SECaaS paradigm.
- **Security and Privacy Group:** This group consists of components that model and implement security policies.
- **Risk Assessment Group:** This group comprises the set of components that deliver data-driven risk assessments. These components enable the services that are described in Section 2.4 of this chapter.
- **Compliance Auditing Group:** This group comprises the set of components that enable and implement the compliance auditing services of SecureIoT-compliant platforms.
- **Programming Support Group:** This group comprises the set of components that support programmers and developers in securing their programs, based

on a set of appropriate annotations that are decoded and operated based on the collected security information.

Furthermore, a key communications' component of the architecture is the data bus, i.e., a communications channel through which real-time data are routed. Platform components may subscribe to the data bus to receive data of interest to them. Following paragraphs provide more information for each one of the above groups.

2.2.2 The Data Management Group

The components of this group are responsible for collecting data from IoT assets, systems, and applications. The data are streamed to the Observations repository of the Global repository. The main components of the group are:

- **Deployed Probes:** Probes collect data from the target IoT systems or applications and stream them to a cloud platform that follows the architecture. The routing is performed by the data routing components.
- **Probe Management and Configuration:** This component is responsible for managing and configuring the deployed Probes. It interacts with the SPEP (Security Policy Enforcement Point), which provides automatic Probe configuration commands that are used to configure the managed Probes. Manual Probe configuration commands may also be issued by security operators.
- **Probe Registry:** This is a registry that maintains a record of the deployed Probes. Probe deployment data, as well as state and configuration data, are maintained by the registry. The registry provides Probe creation, reconfiguration, and search capabilities.
- **SPEP (Security Policy Enforcement Point):** It executes actuating functions towards the IoT system or application under protection. Examples of supported actuating functions include: (i) The automatic (re)configuration of Probes, which can be triggered by security data analysis performed in the Analytics Engine; and (ii) Manual (re)configuration of Probes and other components triggered by the Risk Assessment service, i.e., as a risk mitigation action.

2.2.3 The Analytics Group

SecureIoT provides data-driven security services, and hence, data analytics functionalities are at the core of the architecture. They are provided and implemented as part of the Analytics Group, which includes the following components:

- **Analytics engine:** Provides security analytics functionalities based on the monitored security data that are collected by the Probes, as well as based on

training data and templates. The Analytics Engine trains itself using historical data that are stored in the Observations part of the Global Repository. The outcome of the training is a set of templates that are stored in the Security Template Database. The templates are executed by the Engine and provide the data-driven logic that enables real-time identification of security issues based on the analysis of real-time data that are streamed from the deployed Probes through the data bus. The Analytics Engine raises alerts for security issues that may be detected while monitoring the IoT systems. The generated alerts can be intercepted either by the SPEP component or by the Risk Assessment Engine. Depending on the defined reconfiguration policies, they may carry reconfiguration commands that are to be applied to the deployed Probes. The alerts that are intercepted by the SPEP result in the reconfiguration of Probes, which can be put into effect through the Management and Configuration component. Alternatively, Probe reconfigurations can be executed manually through the Management and Configuration dashboard.

- **Security Template Extraction:** This component trains models based on historical data and extracts reusable templates. In particular, the training part of the analytics algorithm results in the generation of the security templates that are stored into the Security Templates Database and are subsequently used by the monitoring part of the analytics algorithm (e.g., in order to generate alerts or trigger risk assessments).
- **Security Templates Database:** Stores the Security Templates, i.e., pretrained and ready to use machine learning and other data mining models. The templates mechanism boosts reusability and modularity in handling many different security risks.

2.2.4 The Global Repository

The Global Repository comprises various data stores that enable the persistent storage of the data that are used by the other components of the architecture. The following repositories of security data and metadata are included in the Global Repository:

- **Probe Registry:** As already outlined, it contains a record of the deployed Probes along with their attributes, including state information. It facilitates the automatic deployment of Probes and their dynamic discovery.
- **Observations:** This repository contains historic security data that have been collected by the deployed Probes. These data are used by the Analytics Engine to train itself and produce security templates.
- **Analytics Registry:** This repository comprises information about the analytics algorithms that are employed by the SecureIoT platform, along with the

(meta)data needed to configure their lifecycle. Hence, the Analytics Registry is for analytics algorithms what the Probe Registry is for Probes.

- **Policy Repository:** This repository stores the policies that prescribe the handling of security issues (e.g., threats, vulnerabilities).
- **SLA repository:** This repository contains SLA reports along with configuration data about SLAs. It interfaces to components of the SLA Group.
- **Security Knowledge Base (SKB) Repository:** This includes publicly available data on known vulnerabilities and risks, and corresponding CTI (Cyber Threat Intelligence) graphs. Its role is to facilitate the look up and resolution of attack patterns, as a means of automating processes and tasks for risk assessment and compliance auditing services. In the scope of the implementation of the SecureIoT architecture, the contents of the SKB integrate various sources of security knowledge from MITRE (Massachusetts Institute of Technology Research & Engineering) and NIST (National Institute of Standards and Technology), e.g., CAPEC (Common Attack Pattern Enumeration and Classification collection), CVE (Common Vulnerabilities and Exposures collection), CWE (Common Weakness Enumeration collection), CPE (Common Platform Enumeration collection) and NIST's CVSS (Common Vulnerability Scoring System collection).
- **Configuration Management Database (CMDB):** This database contains information about all the assets of the IoT systems that are to be protected, along with their attributes and configuration parameters.

2.2.5 The Security and Privacy Group

This group of components models and keeps track of the different security and privacy policies that are used by SecureIoT-compliant systems. It provides functionalities for modeling policies and trustworthiness in abstract, standards-based and interoperable ways. The group comprises two main modules:

- **Security and Privacy Policies Interoperability:** This component provides support for the definition of interoperable security policies, which are accordingly used by other components of the SecureIoT architecture. To this end, it supports for policy abstractions and policy descriptions in a standards-based manner. For example, in the implementation of the SecureIoT platform, XACML (eXtensible Access Control Markup Language)-based policies are supported in order to facilitate the assessment of attributes and rules in a common abstraction.
- **The Trustworthiness Metrics and Utility Calculation:** This component identifies the trustworthiness of the provider of a certain IoT platform

founded on standards-based metrics specified by the Industrial Internet Consortium (IIC). It encompasses the IIC model and utility calculation tools that facilitate the assessment of a participant's trustworthiness based on defined policies and configuration data.

2.2.6 The Risk Assessment Service Group

This group provides a service that checks the IoT system/deployment that is being protected against a set of known vulnerabilities databases, like NIST's NVD (National Vulnerability Database) or the CVSS (Common Vulnerability Scoring System). The service operates on collected security data for the target IoT deployment as well as stored CTI (Cyber Threat Intelligence) data. In principle, the Risk Assessment service tries to match patterns of vulnerabilities against the collected data and in case it discovers that the IoT deployment is exposed to a vulnerability it raises an alert. To this end, the service leverages the Security Knowledge Base of the Global Repository group. The Risk Assessment group service comprises the following components:

- **Risk Assessment Engine (RAE):** This engine implements the logic of the service itself. It makes use of a Risk Assessment database that contains sets of known vulnerabilities and interacts with its user through the Risk Assessment Dashboard. The RAE comprises its own database (RAE DB) and operates based on its own set of rules (RA rules) that implement the risk assessment logic.
- **Risk Assessment Dashboard:** This component provides a user interface to the Risk Assessment Engine. Mitigation actions that are proposed by the Risk Assessment Engine are presented to the Dashboard allowing the user to decide for their enforcement. These actions are eventually carried out through the SPEP component of the Data Management Group.

The operation of the RAE and of the dashboard based on the SecureIoT data model is illustrated in Section 2.4 of this chapter.

2.2.7 The Compliance Auditing Service (CAS) Group

This group checks and verifies the compliance of IoT system/deployment to regulatory-related requirements, e.g., GDPR, NIS (Network Information System), etc. The service operates on collected security data as well as on regulatory requirements that are expressed as security and privacy policies. The engine that implements the service verifies that the collected data conform to the regulatory requirements as expressed through the respective policies. The latter are managed

through the Policy Engine of the SecureIoT architecture. If no verification can be secured, it raises alerts to the CAS console. The group comprises the following components:

- **Compliance Auditing Manager:** Supports the auditing of the target IoT application, the specification of policies and the generation of the respective (compliance auditing) reports.
- **Compliance Auditing GUI (Graphical User Interface):** This provides a user interface to the Compliance Auditing Manager.

2.2.8 The Programming Support Group

This group implements the Programming Support Service, which provides programming support for security controls through annotations and a set of policies for backing their handling at runtime. The Policy Engine of the SecureIoT architecture is responsible for applying the defined policies and is common to the Compliance Auditing and the Programming Support services. The main component of the group is the Programming Support Manager, which comprises the Policy Manager that allows the specification of policies and the Model Manager that enables the definition of the policy models.

2.2.9 The SLA Group

This group of components supports the management and auditing of Service Level Agreements (SLAs) between IoT systems operators and SecureIoT services providers. To support the management of such SLAs, the group includes the following components:

- **SLA Probes:** These are specialized Probes that collect SLA-related information. They are placed in different parts of the SecureIoT architecture, including Probes in the Data Management, Analytics, and Risk Assessment Groups. The Probes are marked as "SLA Probes" in the architecture figure. They enable the acquisition of (raw) data that are sent to the data routing component. The latter transforms raw data to properly structured Observations and stores them to the Observations Repository.
- **SLA Processing Engine:** This engine processes raw data and generates SLA reports, i.e., reports that provide information about the execution of the SLA (e.g., data collected, data processed, risk assessment reports generated, etc.). The latter are stored in a dedicated part of the Global Repository, i.e., the SLA Repository, which is destined to support SLA Management. The SLA Repository hosts SLA reports along with configuration data about each SLA.

- **SLA Admin Console:** This console provides the means for configuring and managing SLAs, based on the processing of configuration data of the SLA Repository.

A more detailed presentation of the SecureIoT and its use in supporting data-driven security scenarios is available in [5] and [12]. The previous paragraph aimed at outlining the main components and their relationships between them, as the latter have driven the design and implementation of the data model for structuring security data about IoT assets, but also for enabling the configuration of the various Probes that comprise SecureIoT-compliant platforms.

2.3 Data Modelling for Security Systems Interoperability and Configurability

2.3.1 Overview

To support the security data flow and exchange across the different components of the SecureIoT architecture, a common data model has been specified. The model is XML-based and comprises the semantics needed for structuring and storing security data, along with metadata regarding the configuration of a SecureIoT compliant platform. It is meant to be a lightweight model that is tailored to the needs of data-driven IoT applications. Nevertheless, based on the Security Knowledge Base (SKB) Repository, the data model can be linked to conventional sources of security knowledge, as a means of automating risk assessment and cyber threat intelligence processes.

In practice, the model is structured as a collection of interconnected and interrelated schemas, which can be considered as stand-alone data models. In particular, the SecureIoT Data Models form six main logical groups, which are driven by corresponding groups of components in the SecureIoT architecture. These include the following groups:

- **Data Management,** which focuses on the modeling of the data collection and data routing functionalities. These are integral functionalities of any security data collection platform that collects information from multiple sources and Probes, and routes them to different consuming applications (e.g., such as low-latency edge analytics applications).
- **Analytics,** which focuses on the Analytics configuration and management at the Security Intelligence layer.
- **Knowledge Base,** which focuses on modeling of Security Knowledge from third-party standardized sources.

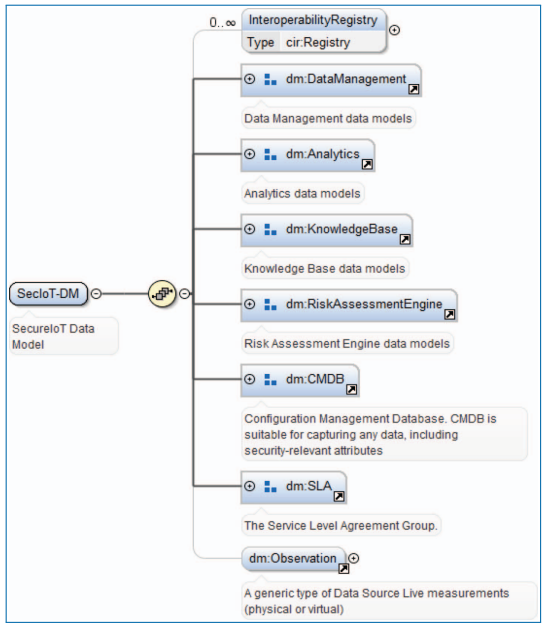


Figure 2.2. Logical groups of the SecureIoT data models.

- **Risk Assessment,** which focuses on the configuration and management of the Risks Assessment and Mitigations SECaaS service. The latter service is illustrated in the following chapter.
- **Configuration and Management DB (CMDB),** which focuses on configuration aspects of the SecureIoT compliant data security infrastructure. It includes modeling of information about the configuration of the monitored IoT systems.
- **SLA,** which comprises data models for configuring and reporting the SLAs.

Some of the Logical Groups are focusing on different aspects of the SecureIoT infrastructure and some are shared along the different layers. All together they form an integrated solution, which is used in the SecureIoT Infrastructure. Figure 2.2 shows the SecureIoT Root Entity of the data model with the different Logical Groups.

2.3.2 Modelling Security Data: The Observation Entity

The Observation data model has a prominent position among the other Logical Groups, given that it is used as a common format for exchanging and reporting data within systems compliant to the SecureIoT architecture. The structure of this sub-model is depicted in Figure 2.3. As evident from the figure, the Observation

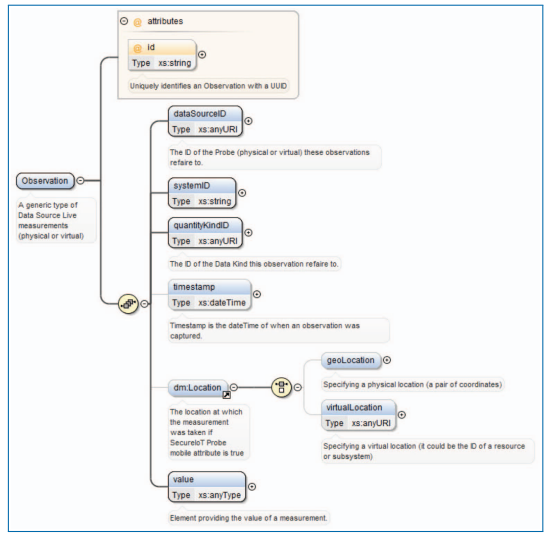

attributes
id
Type xs:string
Uniquely identifies an Observation with a UUID
Observation
A generic type of Data Source Live measurements (physical or virtual)
dataSourceID
Type xs:anyURI
The ID of the Probe (physical or virtual) these observations refaire to.
systemID
Type xs:string
quantityKindID
Type xs:anyURI
The ID of the Data Kind this observation refaire to.
timestamp
Type xs:dateTime
Timestamp is the dateTime of when an observation was captured.
dm:Location
The location at which the measurement was taken if SecureIoT Probe mobile attribute is true
geoLocation
Specifying a physical location (a pair of coordinates)
virtualLocation
Type xs:anyURI
Specifying a virtual location (it could be the ID of a resource or subsystem)
value
Type xs:anyType
Element providing the value of a measurement.

Figure 2.3. Observation entity: modelling security data.

sub-model comprises lightweight semantics tailored to IoT systems. It provides the mean for representing timestamped values for various parameters that can be provided by an IoT system or device. An observation can provide information about different structures of IoT systems, which comprise one or more data sources (e.g., sensor streams, databases). Furthermore, an Observation can be associated with a physical or a virtual location. A physical location is a real-life geolocation represented by coordinates, while a virtual location allows the association of the Observation relative to the location of another system.

In terms of data semantics, each Observation is associated with measurable quantity (i.e., a metric) that is represented based on QuantityKind entity in the model. The QuantityKind is an abstract classifier that represents the type of the quantity that is measured (e.g., temperature, speed, acceleration, power consumption, vibration, etc.). Note that any given Probe can collect data of a given kind, i.e., different metrics are provided by different Probes. Moreover, each Probe is associated to a Data Source from which data originate and a Data Destination to which data are fed. This Data Source is also modeled in the Observation schema. Overall, the Observation schema allows the modeling of 'live' measurements from IoT systems, which may comprise one or more physical and/or virtual data sources.

2.3.3 Configuring and Managing the Data Collection and Routing Process: The Data Management Group

The Data Management Group models the configuration of the data collection system, which facilitates the implementation of data routing functions. The main

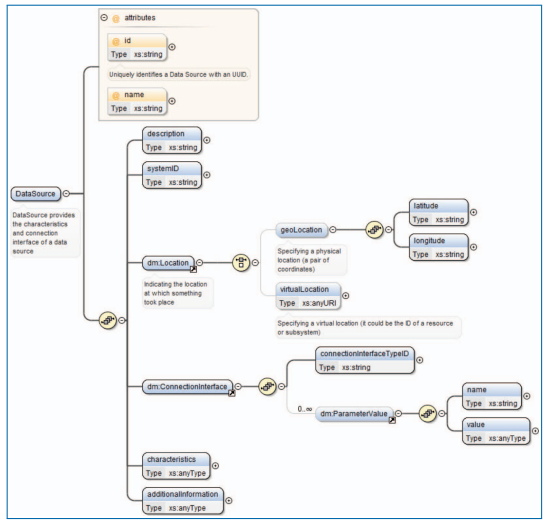


Figure 2.4. Data source entity modelling.

entities are the Probe, which is responsible for collecting data and streaming it to the processing components of the platform and the Observation, which represents the collected datum. Each Observation concerns a metric that is measured, and this is represented as the QuantityKind in the model. A given Probe collects data of a given kind, and if different metrics are to be measured, then different Probes must be deployed. Moreover, each Probe is associated to a Data Source from which data originate and a Data Destination to which data are fed. The structure of a Data Source sub-model is depicted in Figure 2.4. Overall, the data management group comprises the following core entities:

- **Quantity Kind:** Represents the quantity measured or provided by the data source.
- **Connection Interface Type:** Models the type of connection interface which is used at the Data Source and Data Destination entities.
- **Data Source:** Provides the characteristics and connection interface of a data source.
- **Data Destination:** Provides the characteristics and connection interface of a data destination.
- **Probe Type:** Models the type/category of each Probe. Each Probe is classified to a certain Probe type.
- **Probe:** It models and represents a Probe instance. The Probe entity contains the identity and configuration information of the Probe, which is typically deployed within the monitored IoT System.

2.3.4 Modelling Security Analytics: The Analytics Group

The Analytics group focuses on the configuration and management of Analytics configuration at the Security Intelligence layer. It consists of the following entities:

- **Processor Definition,** which is used to describe an Analytics algorithm and its parameters. It lists the metadata of the processor and its required parameters. The structure of a Processor Definition sub-model is depicted in Figure 2.5.
- **Processor Manifest,** which describes a specific instantiation of a Processor Definition. It specifies the concrete data sources that are consumed from the processor and the data sources that it produces. It also provides the parameters required to instantiate the Analytics algorithm.

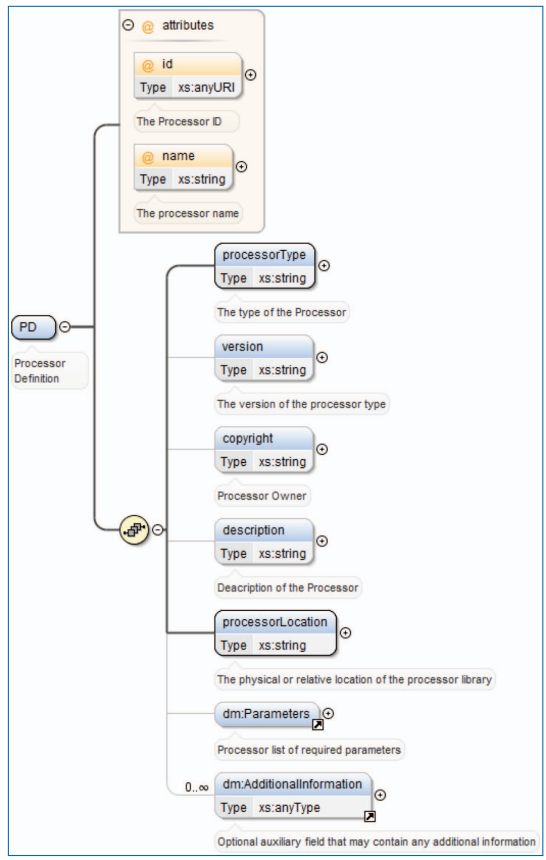


Figure 2.5. Processor definition.

- **Processor Orchestrator,** which groups Processor instances towards creating complex analytics dataflows. The latter can be used to represent various data analytics algorithms, including ML models. Such model can provide additional intelligence to the data-driven security application.

The output of an Analytics workflow is represented based on the Observation format, which includes information about the data source, the system, the timestamp, and the location of the data output through the dataSourceID, systemID, timestamp, and Location elements, respectively. The QuantityKind of the Observation is a STIX compliant element. Its value includes one or more STIX indicators determined by the algorithm. Each STIX indicator contains at least identifiers for the attack, the asset being attacked and (optionally) information about the context of the attack, including links to relevant CVEs.

2.3.5 Security Knowledge Base

The Knowledge Base group models Security Knowledge stemming from available third-party standardized sources. It comprises links to the following schemas:

- CAPEC: MITRE Common Attack Pattern Enumeration and Classification collection.
- CVE: MITRE Common Vulnerabilities and Exposures collection.
- CWE: MITRE Common Weakness Enumeration collection.
- CPE: MITRE Common Platform Enumeration collection.
- CVSS: NIST Common Vulnerability Scoring System collection.

These entities are linked based on standardized schemata provided from each (knowledge provider) organization. These schemata are imported to the SecureIoT Data Models and become available for use by other logical groups.

2.3.6 Modelling for Risk Assessment Services: The Risk Assessment Group

The Risk Assessment Logical Group of the SecureIoT data model focuses on the data entities that support the configuration and management of Risk Assessment and Mitigation Services. The Group is using the CMDB and Knowledge Base entities for the configuration and management of the Engine. The main entities of the group are:

- **Risk Model,** which provides the means for representing a model for a risk assessment service. It provides the means for modeling different risk

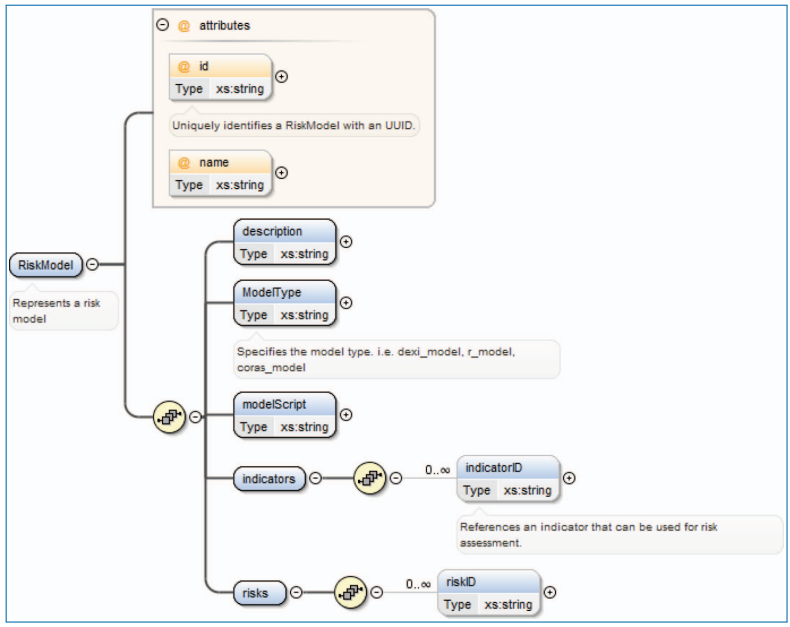


Figure 2.6. Risk modelling for the risk assessment engine.

indicators for different assets. The structure of a Risk Model sub-model is depicted in Figure 2.6.

- **Indicator,** which represents an indicator that can be used for risk assessment.
- **Alarm,** which is used for reporting identified attacks based on the output of an Analytics process.
- **Specific Risk Model,** which represents a specific risk model for users of a specific organization.
- **Risk Plan,** which represents a risk pattern that is associated to a generic risk model that is available to users of all organizations.
- **Risk Assessment Engine,** which models the runtime configuration parameters of the Risk Assessment Engine (RAE).
- **Risk Report,** which models and represents a result of the risk assessment procedure that is delivered to the end user.

2.3.7 Configuring the Data-driven Security System: The Configuration Management Database

The CMDB logical group focuses on the configuration aspects of the data-driven security infrastructure. It comprises information about the monitored IoT systems and the attack scenarios against them. The group consists of the following core entities:

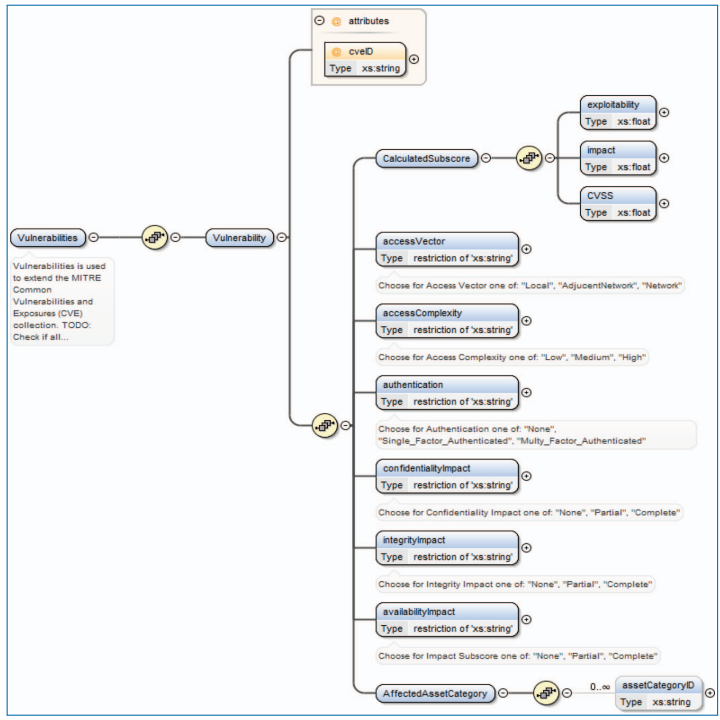


Figure 2.7. Vulnerability modelling in the scope of the CMDB group.

- **System:** Provides a set of observed characteristics for the IoT System.
- **Asset:** Models a cyber, physical, or cyber-physical element within an organization that is used in the frame of business operations.
- **Vendor:** Specifies an Asset's Vendor.
- **Control:** Models a control element that is used to prevent attack impact.
- **Mitigation Plan:** Provides a Mitigation Plan based on a specified Asset.
- **Mitigation Measures:** Provides a list of available Mitigation Measures to be applied for a given Abuse Case.
- **Vulnerabilities:** It is used to extend the MITRE Common Vulnerabilities and Exposures (CVE) collection (see Figure 2.7).
- **Attack Scenarios:** Models a set of Abuse Cases based on the monitored Asset and specified scenarios.

2.3.8 Managing Service Level Agreements (SLA): The SLA Group

The SLA Group comprises data models that support the configuration and reporting of SLAs. The SLA group consists of the following elements:

- **SLA Definition:** Provides the definition of an SLA processor that produces a report about the status of an SLA.
- **SLA Object:** Models an instance of an SLA, which includes specific configuration parameters that enable the auditing and enforcement of the SLA.
- **SLA Report:** Provides the structure of the SLA report. It hosts results calculated by an SLA object.

2.4 Risk Assessment Services

2.4.1 Risk Assessment & Mitigation Service Overview and Components

A Risk Assessment & Mitigation Service (RA&M) has been implemented based on the SecureIoT architecture and the data models that were introduced in earlier sections. Figure 2.8 illustrates the RA&M service implementation, which

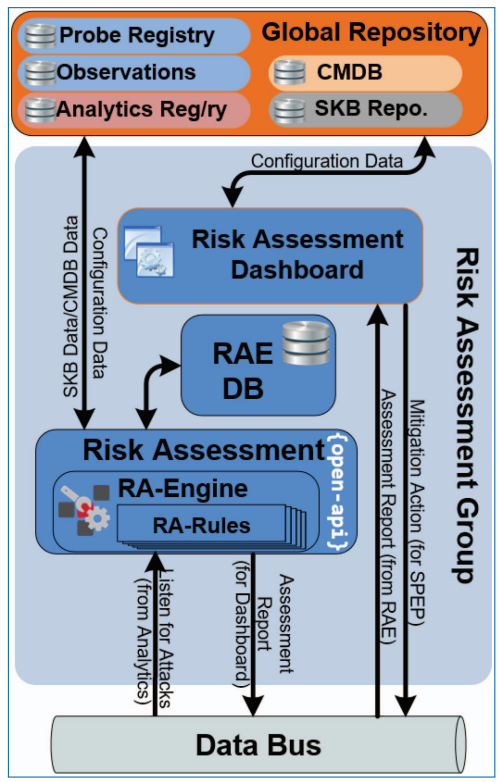


Figure 2.8. Focus of the main building blocks of the risk assessment group in SecureIoT architecture.

leverages components of the SecureIoT architecture. The core component of the RA&M service is the Risk Assessment Engine (RAE), which is responsible for collecting and exploiting data from various sources of the SecureIoT data collection infrastructure in order to assess potential risks and propose mitigation actions. To this end, it comprises a rule engine that enables the execution of Risk Assessment rules, which are executed over data and information from Data Monitoring and Analytics components. Specifically, the data used by the RAE are retrieved either by subscribing to data streams produced by the Probes or by accessing the SecureIoT data storage repositories where the outcomes of data Analytics are persisted. Moreover, the RAE leverages information from the Security Knowledge Base (SKB) where the NIST's Common Vulnerability Scoring System (CVSS) can be used to drive the computation of a vulnerability's "likelihood" factor.

The results produced from the RA engine are presented to end users [e.g., security officers, CERTs (Computer Emergency Response Teams)] through a proper dashboard. The presentation of RA results includes recommendations for risk alleviation activities. For example, they may include recommendations for the enforcement of proper security policies based on the capabilities specified by the monitored platform and the SecureIoT Policy Enforcement Point (SPEP) component. As soon as a mitigation action is specified, the RA engine can send the command to the interface of Policy Enforcement component at the Data Management group. Accordingly, the SecureIoT Policy Enforcement component will execute the action based on the capabilities of the monitored platform.

The RA&M services make use of information from the Monitoring and Analytics component. The Risk Assessment (RA) service monitors the collected data and makes use of the information stored in the Security Knowledge Base in order to perform the risk assessment. Likewise, the Risk Assessment service calculates different configurations and risk criticality values as part of its risk assessment output. The results produced by the RA component are presented to the service operator's dashboard and are accompanied with risk alleviation recommendations like enforcement of proper security policies. As soon as a mitigation action is specified, the RA engine sends the reconfiguration commands to the SPEP at the Data Management group, which enforces the action to the components of the monitoring platform.

The RA can assess risks for services that span multiple IoT platforms (e.g., supply chain services). In particular, the RA&M service supports cross-platform applications by processing security data that are collected from Probes in different IoT platforms. The processing takes advantage of the fact that the platform supports the presented data models.

2.4.2 Risk Assessment & Mitigation Implementation Scenario

One of the use cases that the RA&MS component is used is in the automotive industry and, more specifically, in a connected car use case. The connected car usage scenarios focus on an area within modern IoT such that modern vehicles are no longer just a set of wheels, a body of metal and an engine, but are now full-fledged IoT devices capable of partial/semi-autonomous behavior.

One connected car scenario is where an application involves analyzing vehicle data to assess driver behavior and hence determine the driver risks in order to tailor insurance premiums for customers using this service. The OBU (On Board Unit) obtains raw vehicle data from two sources (Multi-CAN and GPS) and puts context to that information (i.e., how frequent the data are acquired). Collected data are then aggregated, stored locally, and uploaded to the IoT Cloud platform through cellular or WiFi communication interfaces (considering the most appropriate or efficient transmission strategy).

Possible Security gaps and risks for this scenario are:

- Vulnerabilities in software implementation in IoT platform components and vehicle OBU.
- Bad configuration of communication protocols and networks, i.e., not use of HTTPS/encrypted communications.
- Vehicle impersonation (Man in the Middle)
- Tampering of data in transit
- Denial of service at the vehicle (DoS)

2.4.3 Modelling of Security Information Flows and Reports

In order to support the abovementioned scenario, the Risk Assessment & Mitigation services utilizes most of the SecureIoT infrastructure. More specifically (as we can see in Figure 2.1 above of the general SecureIoT architecture and more specifically in Figure 2.8 above focusing on the RA&MS), the following components are used:

- **SecureIoT Probes:** for pushing collected raw measurements to SecureIoT Infrastructure
- **Data Routing:** for storing the data pushed from the Probes to the Global Repository (Elasticsearch)
- **Security Template Extraction:** for training the Analytics algorithms with annotated historical data coming from the IoT Systems

- **Analytics Engine:** which is using the trained templates to analyze the real-time data coming from the IoT Systems and are stored to the SecureIoT Global Repository (Elasticsearch)
- **Data Bus:** which is used as a messaging channel implementing publish/subscribe for the Analytics reports
- **CMDB:** where specific use case data describing the involved Assets and the potential vulnerabilities/threats are stored
- **Risk Assessment Engine:** which is analyzing the published reports from the Analytics Engine to the Data Bus
- **Risk Assessment Dashboard:** which is responsible to visualize the risk assessment reports with possible mitigation actions.

The Risk Assessment scenario set up is divided in two different phases:

1. The possibility to represent and protect the needs of the use cases and the validation using SecureIoT by transferring the data from the use cases to the RAE using communication channels of the project.
2. Its different processing components to transform the data from the Probes.

Regarding the first phase, we list all the inputs required in order to start working in the RAE: a tool for modeling the risk scenarios and transform the model to computer data where in collaboration with the use case owner the security requirements are collected for the threat scenario. With this procedure, we collect information about users, assets, threats, attacks, impact in the system, etc., which we provide as sample below. Once the information is compiled, the modeling of the system is conducted with the help of the modeling tool.

For the Risk Assessment data flow, the direct related components are the CMDB, Global Repository (Observations), Analytics Engine, Data Bus, Risk Assessment Engine and Risk Assessment Dashboard. Figure 2.9 below depicts a high-level dataflow of the validation runtime. More specifically, we can see that first we instantiate the Risk Assessment Engine, which retrieves all the configuration data from the CMDB to be initialized (not depicted in the sequence diagram), and start listening for the Analytics Engine reports by subscribing to a Data Bus topic named after the ID of the Analytics Engine. Then, we set up the Analytics Engine to consume incoming data from the Elasticsearch (Global Repository) where the captured Raw Data are stored. The data are then analyzed for specific Attacks. When an attack is identified, it is pushed to the Data Bus by using as topic name the ID of the instantiated Analytics Algorithm. The report is then picked up from the Risk Assessment Engine which applies preconfigured rules for the specific scenario and produces a

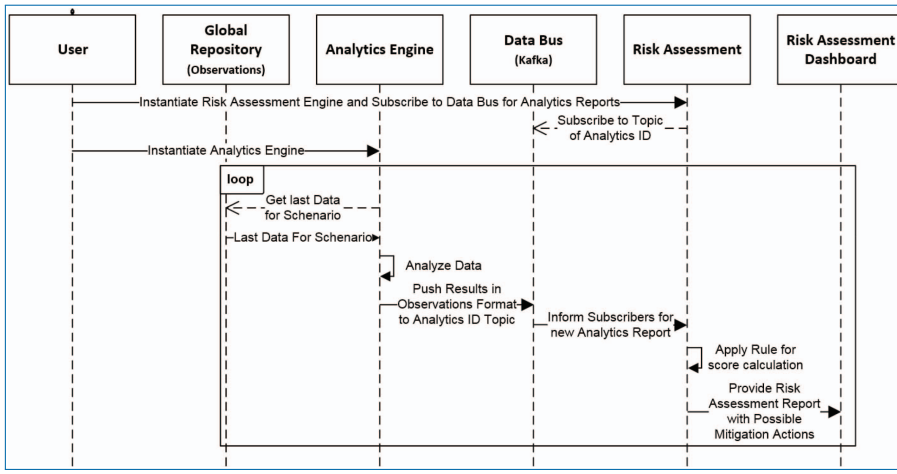

User
Global Repository (Observations)
Analytics Engine
Data Bus (Kafka)
Risk Assessment
Risk Assessment Dashboard
Instantiate Risk Assessment Engine and Subscribe to Data Bus for Analytics Reports
Subscribe to Topic of Analytics ID
Instantiate Analytics Engine
loop
Get last Data for Schenario
Last Data For Schenario
Analyze Data
Push Results in Observations Format to Analytics ID Topic
Inform Subscribers for new Analytics Report
Apply Rule for score calculation
Provide Risk Assessment Report with Possible Mitigation Actions

Figure 2.9. Data management/analytics/risk assessment validation high level sequence diagram.

report with possible mitigation actions. This report is then picked up from the Risk Assessment Dashboard in order to provide it to the end user.

2.4.3.1 Risk Analysis

In order to perform the risk analysis, the first thing to do is to model the use-case scenarios. The scenarios define how the system works, relations with external elements, assets, roles involved, etc. Therefore, it is very important, in order to detect the risks, to know what the normal case of working is, what are the possible cybersecurity problems that can exist, and what are their impacts in the system. Regarding this last one, it is important to describe the usability impact (e.g., service, information stored, etc.) and the economic impact (e.g., inability to provide its service for a whole day would have a huge impact in its business, etc.).

Therefore, in order to be able to perform the risk analysis for the SecureIoT scenarios, we defined a process. The process is composed of the following phases:

- **Attack scenario description:** In the first phase, we discuss about the scenarios of the use case in order to understand what the normal way of working is and what the attack scenarios are. These scenarios disrupt the normal activity and could make the system not to work as intended, data to be stolen, etc.
- **Definition of assets and impact from the technical and business point of view:** After having collected the scenarios, we defined what are their assets and the impact the attacks could have on them. This was studied from a use-case user and cybersecurity expert perspective so we could give ideas of how attacks could impact their services and business in a way they didn't think of.

- **Modeling of the scenarios:** Using the CORAS tool [13], we modeled different scenarios together with risks, assets, attacks, indicators, cybersecurity properties, etc. Following we will examine the models created for each use case in their own sub-sections.
- **Creation of different models and data that are used as input for the Risk Analysis Engine:** Once the models are created, we extract information from them to generate different data files that are necessary for the Risk Assessment Engine. The files we need to generate are the indicators of the cybersecurity threats of each scenario, mitigation actions, risks, and the DEXi model [14]. This last one is a mathematical model that is used for calculating the probability for a specific risk to happen.
- **Analysis:** Finally, we perform the analysis of the model, and the result is shown in the Risk Assessment Engine on the one hand and in a console on the other hand. We followed this approach so we can check the information in both platforms and in the next iteration create our own interface that is more IoT-oriented and provide more useful information of the one coming from the Probes and assurance system.

2.4.3.2 SecureIoT Required Configuration Entities

This section shows how some of the most important entities of the SecureIoT Configuration & Management Database is populated in order to support the above-mentioned scenario.

System

System entity is an abstraction of an IoT System in the real world. The user needs to provide a unique name for the System, a textual description, an organization identifier, information on the system's virtual or physical location and optionally additional information. A System sample can be seen in the Table 2.1.

Table 2.1. Connected vehicle system.


```
{
 "name": "SecureIoT CAD pilot FIWARE cluster",
 "description": "Cloud and car system for SecureIoT testing",
 "location": {
     "geoLocation": {
         "latitude": "41.2533139",
         "longitude": "1.5521598"
     },
     "virtualLocation":"https://cad-pilot.secureiot.eu:8443/"
 },
 "organization": "IDIADA",
 "additionalInformation": {
     "0": "Datum0",
```

```
    "1": "Datum1",
    "2": "Datum3"
  }
}
```


Asset

Asset entity is an abstraction of an Asset belonging to the previous IoT System in the real world. The user needs to provide, among other information, a unique name for the Asset, the IP address and port through which it can be remotely accessed, and the CMDB identifier of the System entity it belongs to. An Asset sample can be seen in the Table 2.2.

Table 2.2. Connected vehicle asset.


```
{
    "name": "OBU IDAPT 001",
    "systemID": "1bdfc95f−2196−4150−91d7−b9fc9d7a7134",
    "probeID": [
        "obu_idapt_001_canbeat",
        "cad_fiware_contextbrokerbeat"
    ],
    "description": "Car",
    "longDescription": "SecureIoT CAD pilot FIWARE cluster",
    "ip_address": "192.168.3.65",
    "port": 3333,
    "location": {
        "geoLocation": {
            "latitude": "51.507879",
            "longitude": "−0.087732"
        },
        "virtualLocation": "https://192.168.3.65:3333/car"
    },
    "securityData": {
        "capecID": [
            "154 Resource Location Spoofing",
            "212 Functionality Misuse",
            "441 Malicious Logic Insertion"
        ],
        "cpeID":
        ["cpe:2.3:a:1024cms:1024_cms:1.4.1:*:*:*:*:*:*:*"],
        "cveID": ["CVE−2017−12697"],
        "cweID": ["Intercept sensitive information
        when the client connects to the server − (200)"]
    },
    "additionalInformation": [{
        "0": "Datum0",
        "1": "Datum1",
        "2": "Datum3"
    }]
}
```

Abuse Case

Abuse Case entity is an abstraction of an attack scenario that may affect an Asset. The user needs to provide, among other information, a unique name for the Abuse Case, the CMDB identifier of the vulnerable Asset entity, information on the Controls and Mitigation Measure that will be used as countermeasures against the attack, as well as a numerical estimation of the attack severity (impact). An Abuse Case sample can be seen in Table 2.3.

Table 2.3. Connected vehicle system abuse case.


```
{
    "name": "Man in the middle attack",
    "assetIDs": ["6e348dd4−e067−4456−8361−48ef8751b868"],
    "controlIDs": [
        "e14d4ce7−344e−475d−a847−d20c1de168ee",
        "0d8557aa−623f−4cd9−b12e−ef05d782be99",
        "11b0220a−84cf−4eea−8e5d−c6cb6db08efe"
    ],
    "mitigationMeasures": [{
        "extendedDescription": "Use PKI for encrypting the
        data before sending to the server. PKI is recommended
        as it is more secure and allows for better interaction
        with other entities of the system",
        "longDescription": "Use encryption mechanisms using a
        PKI strategy",
        "shortDescription": "IN−C1−M1"
    }],
    "processorOrchestratorID":
            "cb53a12b−3a42−4f9e−871d−b23f2a5481e7",
    "attackID": "8dd92273−573b−49c0−a61d−cde79d92ada8",
    "securityData": {
        "capecID": [
            "154 Resource Location Spoofing",
            "212 Functionality Misuse",
            "441 Malicious Logic Insertion"
        ],
        "cpeID":
        ["cpe:2.3:a:1024cms:1024_cms:1.4.1:*:*:*:*:*:*:*"],
         "cveID": [
            "CVE−2017−14084", "CVE−2017−12697",
            "CVE−2018−17187"
        ],
        "cweID": ["Man in the middle (200)"]
    },
    "impact": {
        "availability": 3,
        "confidentiality": 8,
        "integrity": 8
    }
}
```

2.4.3.3 RA&MS-specific Configuration Entities

This section shows how the data of mitigations, risks, and indicators for each of the use cases are introduced in the Risk Assessment Engine. As aforementioned, the models of each use case describe indicators and risks but this information must be introduced in a machine-oriented language so the Risk Assessment Engine can interpret it and use it for the calculations. Following we describe each of these formats.

Indicators

The indicators are introduced in the system via JSON and using a specific format. An example is shown in Table 2.4.

Table 2.4. Connected vehicle RAE indicator.

```
{
    "mode_of_operation": 1,
    "default_value": "No",
    "motivation": "It has been detected a Man in the Middle
        attack in the communication between the car and server.",
    "data_type": "boolean",
    "means": "alarm",
    "question": "IN−1:Is the communication sent unprotected
        from the car to the server?",
    "rule": "PLUGIN_ID=80000 AND PLUGIN_SID=1",
    "indicator_type": 3,
    "sensor_types": [],
    "id": 17
}
```

All the fields are mandatory except for the types of sensors used. This is done for helping in the testing of the information in the system. Regarding the other fields, they are:

- Mode_of_operation: if the system is up or deactivated
- Default_value: what is the output of the indicator by default
- Motivation: what is the attack identified by this indicator
- Data_type: the type of data
- Means: how the information is sent to the Risk Analysis Engine
- Question: the question defining the indicator
- Rule: what sensor is sending this information
- Indicator_type: the type of indicator (e.g., question, value, etc.)
- Sensor_types: the type of sensor (e.g., SecureIoT Probe)
- Id: the unique identifier of the indicator

These data are processed by the Risk Analysis Engine and linked to other data of the system such as the risks, mitigations, etc. The input validated for the indicators comes from the Probes of SecureIoT. That way the "Rule" needs to identify a valid Probe for linking these data to the correct scenario of risk.

Mitigations

Together with risk analysis, the Risk Assessment Engine is able to provide mitigations for identified risks. Although it is not shown in the models of each use case, we define the information in a computer-oriented file so it can be processed and linked to risks in the tool. In Table 2.5, we show one of the mitigations of the connected cars use case.

Table 2.5. Connected vehicle RAE mitigation action.


```
{
    "short_description": "IN−C1−M1",
    "extended_description": "Use PKI for encrypting the
    data before sending to the server. PKI is recommended as
    it is more secure and allows for better interaction with
    other entities of the system.",
    "detailed_description": "Use encryption mechanisms using
    a PKI strategy.",
    "id": 58
}
```


The explanation of each of the fields is as follows:

- Short_description: the identifier used for linking with threats
- Extended_description: information that will be shown to the user regarding the threat
- Id: unique identifier of the mitigation
- Detailed description: high-level description used for short advice in the dashboard

Risks

The document of risks describes all the risks shown in the models in a computer-oriented way. The objective of this is to allow the Risk Assessment Engine to link the risks to the indicators and mitigations that we introduced previously. In Table 2.6, we show one of the risks we used in the document with a description of the fields.

Table 2.6. Connected vehicle RAE risk.


```
{
    "id": 19,
    "short_description": "WRP11−R1",
```

```
    "detailed_description": "Malicious hacker performs a man
        in the middle attack accessing not encrypted data",
    "sections": [1],
    "mitigation_measures": [
        58,
        60
    ],
    "mode_of_operation": "1",
    "organization": "CarManufacturer1"
}
```


The descriptions of each of the fields used for the risks are:

- Mode_of_operation: indicates if the risk is activated or not for the risk assessment
- Mitigation_measures: the link to the mitigations for this risk
- Detailed_description: a description of the risk
- Short_description: identifier used for linking in the internal database
- Organization: the identifier of the organization having this risk
- Sections: the number of possible linked items
- Id: the unique identifier of the risk.

2.5 Conclusions

This chapter introduced a lightweight model for exchanging and reporting security data about IoT devices, systems, and other assets. The model makes provisions for representing security data for various IoT assets and devices. It also provides the means for modeling the ways these data can be analyzed in order to implement security services for IoT systems such as risk assessment services. In the scope of the chapter, we have presented a concrete example of how this model can be used to support a data-driven IoT risk assessment service. Furthermore, the model can represent metadata about the IoT data collection system, as means of facilitating its (re)configuration. The latter can also serve as basis for implementing actionable intelligence by closing the loop to the field in response to security incidents or even in response to specific outcomes of the risk assessment process. Likewise, the integration of configuration metadata facilitates the configurability and programmability of the IoT security system in dynamic environments.

The use of the data model for the implementation of various security services in different use cases demonstrates that there is merit in the overall data modeling approach. In particular, benefits associated with the speed of the configuration and implementation of the data-driven security services have been observed and

validated. Moreover, the versatility of the model has been proven, given its ability to support diverse data collection scenarios for different use cases. However, the adoption of a lightweight approach comes with some limitations in the expressivity of cyber threat intelligence applications, which can be a setback to higher automation and intelligence. This is a trade-off that should be taken into account by designers and deployers of IoT security systems based on the introduced data model. Furthermore, the extent at which the proposed model is tied to the SecureIoT platform should be investigated as well. There are certain parts of the model that are completely agnostic to the architecture of the SecureIoT platform. This is, for example, the case for the modeling of data sources, Observations, and security analytics functionalities. The modeling of these entities is general purpose and can be used in the scope of virtually any IoT system for security data collection. Nevertheless, other parts of the model are more closely affiliated to the modules and to the structure of the SecureIoT architecture, which renders their customization and use in other platforms more time consuming.

Overall, the chapter introduced a novel approach to modeling security data and IoT security systems configurations. The approach emphasized security data collection and analytics, towards delivering intelligent functionalities for security teams, including data-driven risk assessment and actionable intelligence tied to their outcomes. The approach can be certainly extended and customized to different requirements, thanks to the use of XML-based, extensible and rather general-purpose entities for modeling security data and their analysis.

Acknowledgements

This work has been carried out in the scope of the H2020 SecureIoT project, which is funded by the European Commission in the scope of its H2020 program (contract number 779899). The authors acknowledge valuable help and contributions from all partners of the project.

References

[1] Q. Jing, A. V. Vasilakos, J. Wan, J. Lu, and D. Qiu, "Security of the Internet of Things: perspectives and challenges," in Wireless Networks, vol. 20, issue 8, November 2014, pp. 2481–2501.

[2] R. Martin, S. Schrecker, H. Soroush, J. Molina, J. P. LeBlanc, F. Hirsch, M. Buchheit, A. Ginter, H. Banavara, S. Eswarahally, K. Raman, A. King, Q. Zhang, P. MacKay, and B. Witten. (2016). Industrial Internet Security Framework Technical Report. 10.13140/RG.2.2.28143.23201.

[3] A. Bolster and A. Marshall, "Analytical metric weight generation for multi-domain trust in autonomous underwater MANETs" Proc. IEEE 3rd Underwater Commun. Netw. Conf. pp. 1–5, 2016.
[4] U. Jayasinghe, G. M. Lee, T. Um and Q. Shi, "Machine Learning based Trust Computational Model for IoT Services", in IEEE Transactions on Sustainable Computing, May 2018.
[5] A. Roukounaki, S. Efremidis, J. Soldatos, J. Neises, T. Walloschke, N. Kefalakis, Scalable and Configurable End-to-End Collection and Analysis of IoT Security Data: Towards End-to-End Security in IoT Systems. GIoTS 2019: 1–6.
[6] K. Umezawa, Y. Mishina, S. Wohlgemuth, and K. Takaragi, (2018). Threat Analysis using Vulnerability Databases – Matching Attack Cases to Vulnerability Database by Topic Model Analysis.
[7] H. C. Chen, M. A. Al Faruque, and P. H. Chou, "Security and privacy challenges in IoT-based machine-to-machine collaborative scenarios," 2016 International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS), Pittsburgh, PA, 2016, pp. 1–2.
[8] F. Sadique, and S. Cheung, I. Vakilinia, S. Badsha, and S. Sengupta, (2018). Automated Structured Threat Information Expression (STIX) Document Generation with Privacy Preservation. 10.1109/UEMCON.2018.8796822.
[9] The Common Vulnerability Reporting Framework (CVRF), Version 1.1, available at: https://www.icasi.org/the-common-vulnerability-reporting-framework-cvrf-v1-1/
[10] R. Danyliw, *et al.* "The Incident Object Description Exchange Format." RFC 5070 (2007): 1–92.
[11] T. Takahashi, *et al.* "An Incident Object Description Exchange Format (IODEF) Extension for Structured Cybersecurity Information." RFC 7203 (2014): 1–28.
[12] S. Astaras, S. Efremidis, A.-M. Despotopoulou, J. Soldatos, and N. Kefalakis, "Deep Learning Analytics for IoT Security over a Configurable BigData Platform", In the Proc. of the 22nd International Symposium on Wireless Personal Multimedia Communications, IEEE WPMC, 2019, Lisbon, Portugal, November 2019.
[13] F. Vraalsen, and F. Braber, M. Lund, and K. Stølen, (2005). The CORAS Tool for security risk analysis. 402–405. 10.1007/11429760_30.
[14] M. Bohanec, V. Rajkovič, I. Bratko, B. Zupan, and M. Žnidaršič, DEX methodology: Three decades of qualitative multi-attribute modelling. Informatica 37, 49–54, 2013.

DOI: 10.1561/9781680836837.ch3

Chapter 3

Data-driven IoT Security Using Deep Learning Techniques

By Stefanos Astaras, Nikos Kefalakis, Angela-Maria Despotopoulou and John Soldatos

For nearly two decades, machine learning techniques have been extensively researched in terms of their ability to indicate cybersecurity attacks in the networking and cloud assets of IT systems. To a lesser extent, they have been deployed in cybersecurity for IoT systems. Nevertheless, their potential for detecting attacks in all layers and components of an IoT system has not been adequately investigated. This chapter introduces deep learning techniques for the detection of abnormal behaviors and anomalies in the operation of state-of-the-art IoT systems that comprise smart objects such as connected vehicles and socially assisted robots. The presented approach uses variational autoencoders and has been validated on real-life datasets derived from smart objects with very promising results. Moreover, the introduced algorithms have been integrated within a data-driven IoT security platform, namely the SecureIoT platform that has been presented in a previous chapter. Overall, the present chapter discusses a novel deep learning approach for IoT security, while validating it in realistic datasets other than the legacy openly available datasets that are commonly used by security researchers.

3.1 Introduction

In recent years, internet-connected objects are proliferating in number which is one of the main drivers for the emergence and the rise of the Internet of Things (IoT) computing paradigm. There is also a rising number of the different types of internet-connected objects, which increase the versatility of state-of-the-art IoT applications. Furthermore, we are also witnessing a rising sophistication of internet-connected devices, which is also opening up opportunities for IoT applications with advanced functionalities. This is the case with smart (connected) objects that can operate autonomously, such as unmanned aerial vehicles (UAVs), autonomous guided vehicles, and socially assisted robots. As a prominent example, autonomous vehicles are set to disrupt the transport sector through enabling driverless cars that will move more safely than human-driven ones. For another example, socially assisted robots make possible new assistive applications for patients and elderly users in need of coaching or support for managing their disease. Nevertheless, this rising sophistication of IoT applications is inevitably followed by new possibilities for compromising their operation based on security attacks. Such attacks can occur at multiple levels and components of an IoT application, including attacks against cloud components, edge devices, networking infrastructures or even the smart objects and IoT devices themselves. Thus, there is also a need for novel cyber-security mechanisms that could safeguard the operation of IoT applications against adversarial attacks by malicious parties [1, 2].

One of the most prominent ways for implementing cyber-defense mechanisms for IoT systems entails the collection and analysis of data from their various components. Indeed, reference architectures for IoT security such as the Industrial Internet Security Framework (IISF) [3] specify security workflows that involve the following steps:

- **Monitoring:** The continuous collection of security-related data from the various assets of an IoT system through appropriate probes.
- **Analyzing:** The analysis of the data based on data analytics and data mining algorithms. The aim of the analysis is in most cases to discern indicators of security risks, vulnerability, and attacks (e.g., an attack pattern).
- **Act:** The implementation of remedial measures and actions based on actuation functions.

In-line with these workflows, the SecureIoT project has implemented a data-driven security platform, which provides the means for collecting security data from various IoT components in scalable and highly efficient (e.g., with low latency) ways. A brief description of the platform is provided in the previous chapter of the

book, while more detailed descriptions are also available in other publications of the authors [2, 4]. Data-driven security platforms like SecureIoT enable the integration of different types of algorithms for analyzing security-related datasets, for the purpose of identifying patterns and events that are interesting from a security viewpoint. In fact, the algorithms used are among the more critical components of the data-driven security systems, as their efficiency determines the overall intelligence of the security mechanism.

For nearly two decades, a great deal of data analytics and machine learning techniques have been researched in terms of their ability to identify security risks, attack patterns, and potentially abnormal behaviors. For example, the authors in [5] present a range of network intrusion detection techniques, including statistical, knowledge-based, and machine-learning approaches. For another example, computational intelligence methods have been also employed [6], including deep learning techniques such as Artificial Neural Network (ANNs). Techniques based in ANNs have also been employed for misuse detection in [7]. The system developed in [7] took advantage of a knowledge base that contained attack signatures. Machine learning techniques have also been put to service for more complex security tasks, such as the identification of botnets [8] and intrusion detection in sophisticated scenarios [9]. Intrusion detection use cases have been also addressed by taking advantage of genetic algorithms [10]. In general, most of the research efforts for applying machine learning in cybersecurity have been focused on techniques for identifying misuse and/or anomaly detection. Their main goal has been to identify known intrusions with high confidence, while at the same time decreasing false-positive rates for the identification of unknown attacks.

Most of the above-listed systems have been focused on network attacks. The emergence of new distributed computing models, like cloud computing, gave rise to the application of similar techniques for different types of IT assets and infrastructures [11]. For example, in [12] supervised machine learning techniques were used for identifying security incidents compromising cloud infrastructures. The application of machine learning for cloud security revealed promising results along with limitations. The use of larger amounts of historical data for training was among the techniques that later works employed in order to remedy those limitations [13].

As already outlined, the emergence of IoT systems and applications introduced new cyber assets, along with a set of new challenges. Machine learning techniques show significant promise for identifying anomalies and attacks associated with IoT assets, yet they have not systematically been studied yet. Deep learning models can be also applied, as IoT devices produce large amounts of data that could be utilized for security analysis. Specifically, deep learning is a class of machine learning graph algorithms where each base input feature vector is classified, clustered, or otherwise modeled in a multi-stage process that performs both higher-order feature

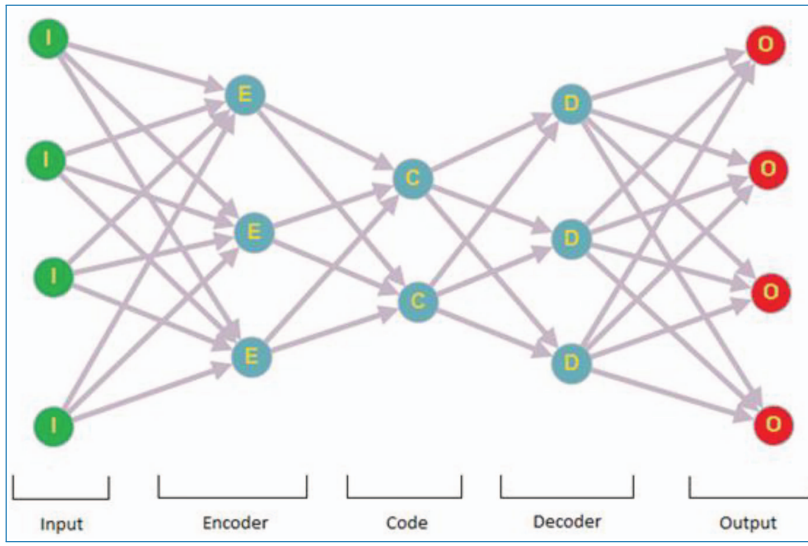


Figure 3.1. Structure of variational autoencoder (VAE).

extraction and class fitting in a unified way. The defining characteristic is that the entire process is simultaneously fitted on the data, and hence, there is no need for a pre-designed feature extraction or pre-processing step. Deep learning models are further divided into numerous sub-classes based on their topology, such as Recurrent Neural Networks (RNNs), Deep Belief Networks (DBNs), and Convolutional Neural Networks (CNNs) [14].

This chapter presents a deep learning approach to anomaly detection in IoT systems, notably systems that comprise smart objects like connected vehicles or robots. Specifically, the chapter focuses on a deep learning method that has been proved capable of detecting systematic faults and intrusions in IoT systems, namely the Variational Autoencoder (VAE) [15]. This deep learning structure has multiple hidden layers that first reduce the dimensionality of the input and then scale up to reproduce it (Figure 3.1). It is conceptually separated into the encoder (i.e., the initial layers that perform dimensionality reduction) and the decoder (i.e., the output layers that recreate the output). In the variational approach, the code layer includes an adaptive probability distribution for the latent variable. The presented approach has been developed and validated based on real-life datasets that have been made available by OEM vendors and IoT integrators in the scope of the H2020 SecureIoT project [2]. Furthermore, the deep learning algorithms introduced on this chapter have been integrated in the SecureIoT platform for data-driven security [4], which has been presented in the previous chapter.

One of the principle innovations of the technique lies in the use of real-life datasets from smart objects for the training and evaluation of the deep

learning models. Most of the above-presented methods have been developed and evaluated based on reference datasets, such as the one used in the scope of the Third International Knowledge Discovery and Data Mining Tools Competition back in 1999 and several datasets with network-level information provided by DARPA [16–18]. The VAE models of this chapter have leveraged true IoT datasets derived from novel smart objects/devices like a socially assisted robot and a connected car. These datasets are therefore described in subsequent sections of the chapter.

The rest of the chapter is structured as follows: Section 3.2 following this introductory section describes the methodology followed and the datasets used. Section 3.3 presents the VAE models developed and used. Section 3.4 presents validation results on IoT real-life datasets. Section 3.5 is the final and concluding section of the chapter.

3.2 Methodology and Datasets

3.2.1 CRISP-DM Methodology

In order to experiment with various VAE models over the available datasets, we followed the Cross Industry Standard Process for Data Mining (CRISP-DM) Model for Knowledge Discovery [19], which entails the following activities and phases:

- **Business Understanding,** which is the starting point of the process and refers to the need for understanding the anomaly detection problem at hand.
- **Data Understanding,** which aims at examining the data in order to gain valuable insights on the applicability of certain data mining schemes. Data understanding enables the identification of data patterns in the datasets, which will serve as a basis for synthesizing candidate machine learning schemes.
- **Data Preparation,** which will transform the maintenance datasets into a format appropriate for employing data mining and machine learning models. This step involves Extract Transform Load (ETL) processes.
- **Modeling,** which discerns proper machine learning or statistics schemes for the anomaly detection problem at hand. It interacts closely with the data preparation phase in order to ensure that the available training datasets are appropriate for fitting the target/identified models.
- **Evaluation,** which entails the assessment of the produced model in terms of efficiency as the latter is reflected in the speed of training, the speed of model execution, its noise tolerance, as well as its expressiveness and explanatory ability. Metrics such as classification accuracy, errors in numeric prediction, lift and conviction measures and more are frequently used.

- **Deployment,** which involves the operationalization of successful models based on the results of their integration in the SecureIoT platform that is described in the previous section.

3.2.2 Connected Cars Dataset

The connected car datasets were provided by IDIADA,[1] one of the most prominent engineering companies providing design, testing, engineering, and homologation services to the automotive industry. They have been provided from a simulation tool that generates vehicle performance data from different travels. There are two data sources: the first one is Controller Area Network (CAN) data, while the V2X module provides the second. The CAN data are metrics used to represent different conflicts that may occur, like how many unexpected data frames have been received during a fixed period of time. On the other hand, the V2X module creates application data of a vehicle such as its speed or its GPS coordinates.

The available data follow the SecureIoT data model, which is described in the previous chapter. Hence, they can be integrated into the SecureIoT platform and its global repository. Overall, there have been put into service 34 datasets with normal behaviors and 3 containing anomalies.

3.2.3 Socially Assistive Robots Datasets

These are data generated by a QT robot [5]. The latter is a robot designed to stimulate the memory functioning and physical activities of elderly people. Moreover, it can be used to increase the efficiency of education by easily attracting the attention of autistic children to teach them new life skills. The available datasets refer to multiple use cases and do not describe the same system's elements. One of the datasets has been generated after the QT robot has tried to recognize emotions of people standing in front of it. The other datasets contain more technical data such as motor positions that describe the robot's behavior and have been obtained by playing a game. In some of the datasets, the robot's partner has the task to make the same gesture as it, while in others the partner of the robot has been assigned to show the correct sequence of cards in a memory game. Data are integrated in the SecureIoT Global repository, and they follow the structure of the SecureIoT Data model.

Additionally, there are data generated from an IoT-cloud platform (namely CloudCare2U [5]), a solution aiming to provide a life as normal as possible to chronic disease patients, based on a 40-day simulation. Five different types of

1. https://www.applusidiada.com/global/en/

datasets can be found, each associated to a set of sensors from the same simulation. They are described as number 1 to 5 in the section about scenarios of socially assistive robots. The first dataset contains environmental sensing data such as temperature or humidity, and the second one provides data from wearable sensing data such as the physical activity of the patient. The third and fourth datasets represent, respectively, visual sensing data, which give information about the patient such as their position or age, and resting furniture data to know if the patient is on the sofa. The last dataset contains measurements of a patient's vitals such as their heartbeat.

3.3 Variational Autoencoders for Anomaly Detection

3.3.1 VAE Architecture

A variational autoencoder, being a deep learning model, processes the input in multiple layers. The first layer is always the input layer, which forwards each input scalar into subsequent nodes. VAEs are feed-forward models, which means that the input layer has to include all the inputs that are relevant for a single output. So, in a sequence of 5 consecutive measurements of 5 different sensors, we need an input layer of length 25 (as is, for example, the case in the CAN application dataset).

Next is the value normalization layer. Each input scalar is offset by its minimum value and scaled down by its new maximum value so that it is always in the range of 0–1. This helps numerical stability and the multi-dimensionality of the model. Without normalization, models tend to be biased towards inputs that naturally take large values. The length of this layer is equal to the input layer.

The encoder part is formed by one or more fully connected layers. Each node in a fully connected layer receives an input from every node of the previous layer (plus a constant bias input), multiplies it by a learned weight, sums the inputs, and feeds the sum to a non-linear activation function (Figure 3.2). The activation function of the node helps the overall model learn the non-linearities of the process. The output of the node is the result of the activation function. Models that perform classification tend to favor sigmoid functions, while regression problems (such as ours) tend to use the rectifier linear unit function (see Figure 3.3).

In the VAE, the consecutive fully connected layers decrease in length, eventually down to the desired code dimension.

In the variational approach, a latent Gaussian probability distribution is formed from the code, using a custom layer with this format:

- Two fully connected layers in parallel. The first produces the mean of the distribution, and the second the variance.

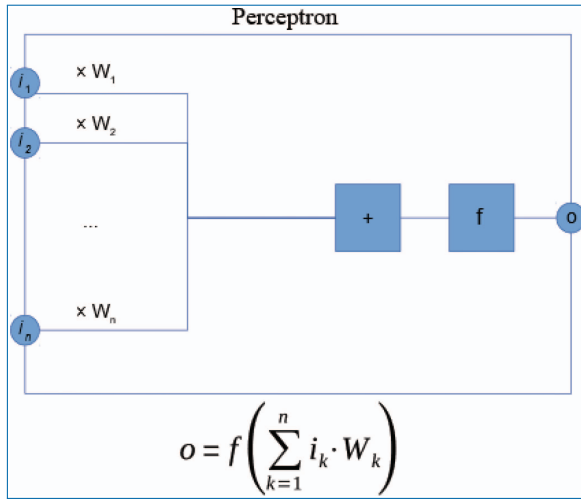


Figure 3.2. The general fully connected node (perceptron).

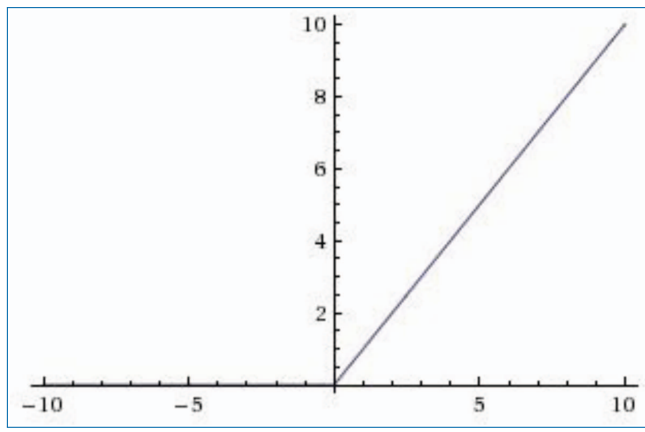


Figure 3.3. The rectifier linear unit function (ReLU) [20].

- A sampling layer that follows the previous two and samples a Gaussian distribution with the parameters it received.

The probability distribution is generally multi-dimensional. In our application, 2D and 3D latent representations work well.

The decoder has the same number of fully connected layers as the encoder, but in reverse; the length of the layers starts from the code dimension and scales up to the number of output nodes (which here is the same as the input nodes). The same ReLU activation function is used.

Finally, a de-normalization layer is applied to the outputs, to bring them back to their original range.

3.3.2 VAE Training

Training a variational autoencoder is efficient, as it follows the conventions of training any deep learning model. There are four steps to designing a training process, which we apply in-line with the CRISP-DM model [19]:

1. Preparation of the dataset
2. Choosing a batch size
3. Choosing an optimizer
4. Choosing a loss function

To prepare the dataset, we form sequences of measurements according to the model we want to train. For example, in a model that takes a sequence of 5 input vectors, we concatenate each vector in the dataset with its neighbors, so that every vector eventually appears in 5 different sequences, at a different position each time. Then, the dataset is shuffled, and 20% is removed and reserved for testing.

The batch size is the number of samples that are passed into the model in parallel. A large batch size leads to faster training but can lead to reduced accuracy [21]. We use a safe batch size range from 32 to 128 samples, since, as a rule of thumb, batch sizes over 256 samples can create problems in the operation of the model.

The optimizer is the learning algorithm that updates the weights of each node in the model. We use the Nadam optimizer (Nesterov-accelerated Adaptive Moment Estimation), which is the Adam (Adaptive Moment Estimation) optimizer that incorporates Nesterov momentum [22]. This algorithm assigns a different learning rate to each parameter and keeps track of recent training updates to better guide the process and mitigate noise.

The important innovation that variational autoencoders provide is the loss function, which is developed in terms of the encoder/decoder/code separation:

$$l_i = -E[\log(p_\varphi(x_i|z))] + KL(q_\theta(z|x_i)||N(z))$$

where:

- x_i is the input with index i.
- z is the latent representation of x_i, i.e., the code that the encoder produces.
- $p_\varphi(x|z)$ is the distribution of the decoder, given a certain code.
- $q_\theta(z|x)$ is the distribution of the encoder, given a certain input.
- $N(z)$ is a distribution we define, usually a unit Gaussian $N(0, 1)$.
- $KL(P\|Q)$ is the Kullback–Leibler divergence. It measures how much information is lost when using distribution P to represent another distribution Q.

The first term of the loss function is the expected log-likelihood that the decoder reconstructs the input accurately. It is zero for a perfect reconstruction and grows

as the model becomes less accurate. We need to use a practical substitute for performance and numerical stability, a common choice being the root of the mean absolute error.

The second term acts as a regularizer. Normally, the encoder can learn very localized, essentially meaningless representations, leading to overfitting. However, the Kullback–Leibler divergence will grow as the encoder's distribution becomes more complex and drifts away from our base distribution $N(z)$.

As such, minimizing this loss function leads to an accurate model via the first term, while it mitigates overfitting and keeps latent representations meaningful via the second term.

3.3.3 Algorithm Fitness

Variational autoencoders are not direct predictors; their primary task is to encode the context efficiently. This facilitates handling general abnormal cases with versatility, making autoencoders a standard tool for detecting intrusions and faults in industrial IoT contexts.

Labels are predicted by applying the model to the input with all possible label values and choosing the one that reproduces the input more accurately (using a distance metric). Because autoencoders focus on correct modeling of the data, missing features or labels can be compensated for by the model itself, thus providing the following advantages:

1. The model can reconstruct missing features. We can artificially replace some features from input vectors with "missing value" placeholders in the training phase, but still include the real intended values in the loss calculation. The model will learn to infer missing values.
2. The model can be fitted to new abnormal cases without retraining. We can resume training with the new data and provide the new label as input. This is especially helpful when transitioning from preliminary training and scaling the dataset.
3. The model can detect abnormalities that weren't encountered before. It cannot specifically identify them, but if an input vector fails to be accurately reproduced in the output with any label, it is evidence of a new use case that was not encountered during training.
4. The model is very efficient on data of low dimensionality and isolated attacks. It can learn the correlations between the features very well and isolated attacks (i.e., attacks that are limited on the number of features they affect) are easily recognizable and do not make the input similar to a normal situation or another use case.

Compared to a pure deep learning discriminator on ideal conditions, variational autoencoders might exhibit efficiency disadvantages. When the entire dataset and all the expected threat kinds are available beforehand, a pure discriminator may be more accurate, because it is trained to model the boundaries of the data labels, which is a simpler task than modeling the data itself. This is especially apparent on broad labels with complex underlying structures. For example, an application may have data organized in multiple clusters, while the normal clusters are far and easily separable from the abnormal ones. An autoencoder will try to learn the entire clustering while a discriminator may be accurate with just the simple boundary.

Our approach focuses on memoryless, feed-forward modeling. A few of the expected attacks are evolving, i.e., cannot be identified by a single input vector, but are only apparent on multiple consecutive measurements. Feed-forward models are unable to detect this type of attack out of the box, but we can aggregate measurements as an extra step and compute temporal information (such as input deltas) to include in the input vectors. This is possible because even the evolving attacks take place over a short duration (relative to sample spacing). This approach ensures better coverage of use cases, while being simpler and more efficient than using a recurrent model.

3.4 Application and Validation Results

We have trained variational autoencoders of various complexities and successfully applied them to the available datasets.

3.4.1 Anomaly Detection in Connected Cars

The IDIADA dataset is a series of measurements from sensors installed in a vehicle (such as fuel, speed, and steering angle) as it is being driven. The abnormal cases are all intrusion-related and fall into the following categories:

- At least one sensor report is being overwritten with a static, usually extreme value (e.g., speed or RPM appear as their maximum values despite the current throttle position).
- At least one sensor report is being overwritten with a seemingly regular value, but the way the value evolves is abnormal (e.g., the vehicle does not appear to properly consume fuel as expected while it is being driven).

Both cases can be handled by a variational autoencoder, as they describe deviations from normal operation. The autoencoder can be fitted to the normal driving

conditions and subsequently detect many kinds of attacks without needing all possible malicious contexts from the beginning.

However, there are some challenges. The normal driving operation is not simple, as the state of the vehicle can vary according to the traffic and road conditions, although we expect the relations between the speed, throttle, RPM, gear, and steering angle to stay the same. In addition, one kind of attack takes place over a duration and cannot be detected from a single time instance. We handle these challenges by introducing data augmentation. Specifically, we resample the dataset so that the various normal driving conditions are equally frequent during training, and we aggregate sensor readings and inject the calculated sensor slopes along their current values.

To validate the performance of the algorithm, we randomly split the post-processed dataset into training (70%) and testing (30%) sets. Because there is no strict confidence measure, we used the maximum squared reconstruction error of the normal case as the classification threshold (i.e., inputs with a reconstruction error on the normal label over a threshold are classified as abnormal). This allows taking advantage of this model's flexibility on abnormal cases and leads to the performance curves depicted in Figures 3.4 and 3.5.

Expectedly, very low and very high thresholds lead to performance equivalent to a random choice, as all test inputs are either rejected as abnormal or accepted as

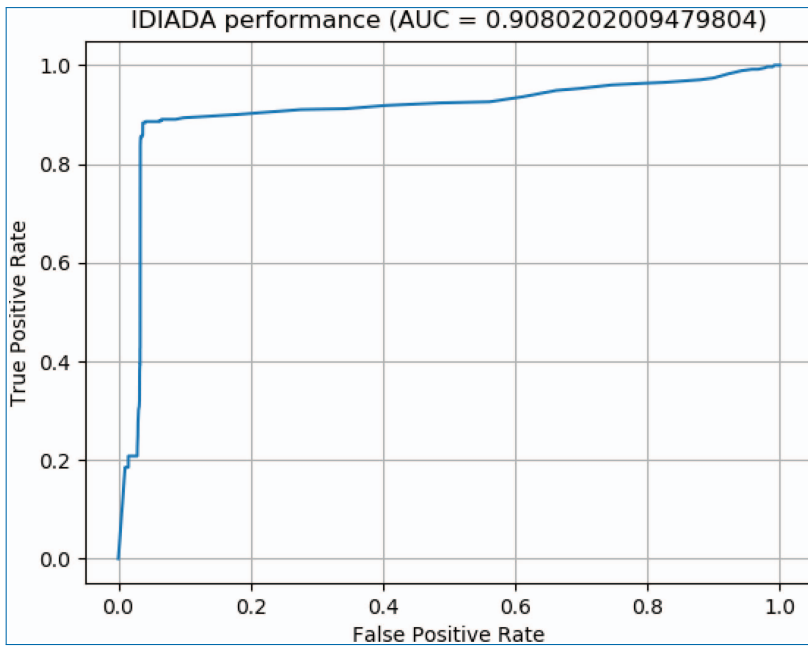


Figure 3.4. ROC curve of the connected car classifier.

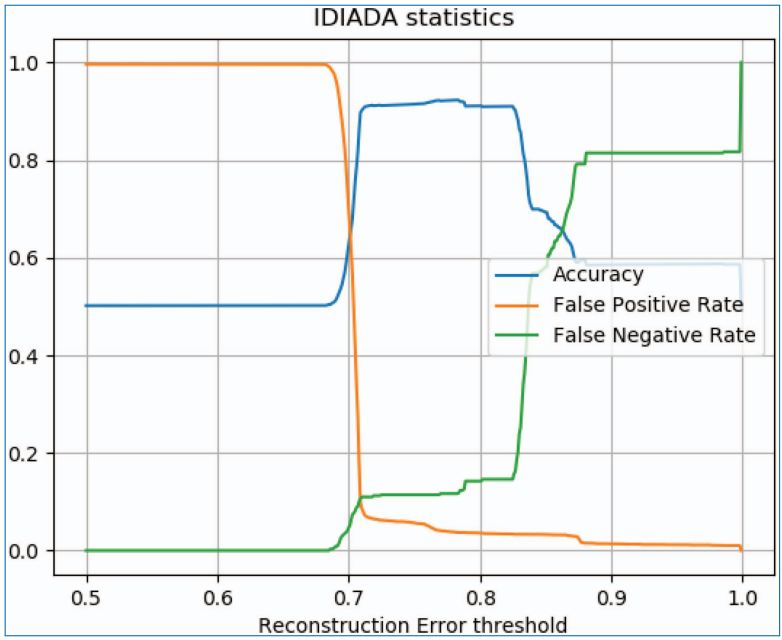


Figure 3.5. Performance variations on the connected car dataset as a function of the reconstruction error threshold.

normal, respectively. The optimal error threshold for this model is $t = 0.76$, where the accuracy score is 92% on this test set.

3.4.2 Anomaly Detection in Socially Assistive Robots Use Cases

3.4.2.1 QT Robot Dataset

The QT Robot dataset contains sensor and context reports. The data are broadly labeled as normal or abnormal and have three components:

- **Motor readings.** The head, shoulders, and elbows are monitored for their angular position and velocity in terms of yaw, pitch, and roll (depending on the degrees of freedom of each motor).
- **Gesture reports.** The type of gesture to be performed by the QT robot, its duration, and whether it was performed normally.
- **Where applicable, event timestamps.** Specifically, the start and end times of the various runs of the QT Memory Game are recorded.

The task is to detect abnormalities in the robot motors with respect to the intended gesture. Although it is hard to discern the gestures from the motor

readings alone, the latter follow a regular pattern dependent on the gesture. The duration also does not vary in the normal cases.

A variational autoencoder is ideal, as it can accurately model the normal behavior because there is little variation among normal values, despite the complexity of the relation. We can also detect all kinds of abnormalities without knowing them beforehand.

Similar to the other use cases, we validate the performance of the algorithm by randomly splitting the post-processed dataset into training (70%) and testing (30%) sets. For this test, the motor position readings were used for the input, and the label ground truth was derived from the gesture reports. The model we tested provided the performance curves of Figures 3.6 and 3.7.

In this case, the ideal error threshold is $t = 0.02$, where the accuracy score is 73%. In general, the optimal threshold is dataset dependent as well as model dependent. Here we can observe that the normal use case is more easily representable, owing to more tightly coupled data, while abnormal data are more sparsely spread, as attacks or faults can transpire in multiple ways. As such, the false-positive rate tends to drop more abruptly than the false-negative rate rises.

3.4.2.2 IoT-cloud Platform (CloudCare2U) Dataset

The CloudCare2U dataset contains body and environment measurements from the CC2U simulator. The data are broadly described as normal or abnormal.

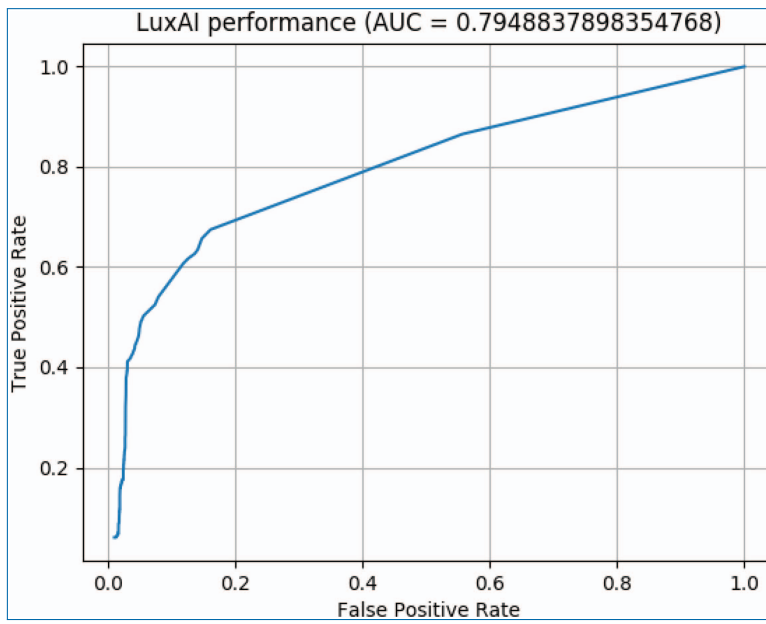


Figure 3.6. ROC curve of the QT robot classifier.

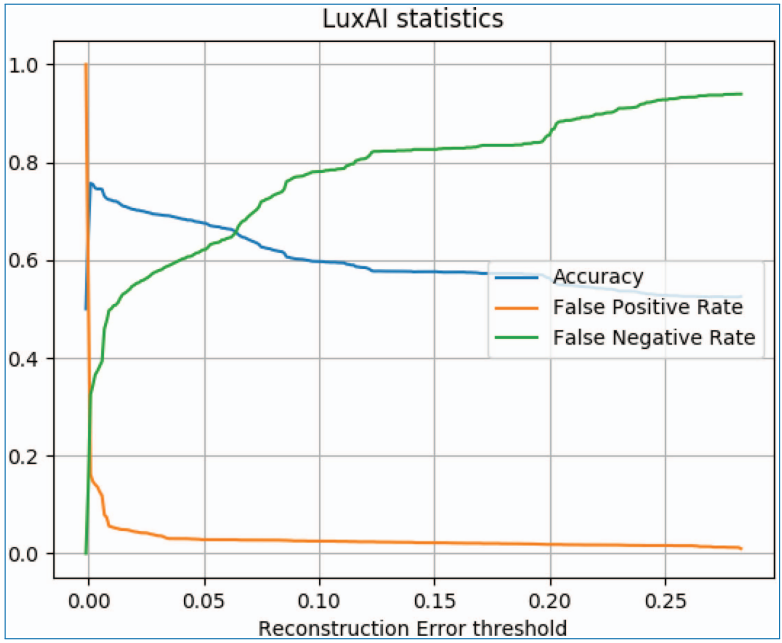


Figure 3.7. Performance variations on the QT robot dataset as a function of the reconstruction error threshold.

However, due to the nature of the intended sensors, the various measurements tend to be correlated with the current timestamp, especially on the normal case. For example, the step counter consistently records two days of low step counts following five days of average step counts, simulating a workday/weekend separation in behavior. The state of the user (sleeping, walking, grooming, etc.) is correlated with the time of day, as do the various home sensors (lightning, illuminance, etc.). The abnormal data, even when the measured values are not out of their specified range, do violate the above assumptions.

Aside from absolute temporal correlations, there is more coupling between the sensors. The user's body state is dictated by their current activity. The temperature, humidity, and gas sensor measurements tend to increase together, and this happens when the user is performing an activity in that room. Following an entertaining activity, the user's reported boredom drops.

A variational autoencoder learns this series of correlations and properly models the normal operation. This means that we do not need to know all the possible abnormalities beforehand and do not need to be restricted with a pure discriminator. Rather, we can expect to detect many different deviations from the normal conditions. Moreover, we get the advantage of flexible training. For example, in a real setting, seasonal changes will change the usual values of the sensors. We do

not need all the data beforehand, the model will operate correctly with the current conditions, and we can adapt it by extending the training as the expected changes are observed.

For this dataset, we followed the same procedure of randomly splitting the post-processed dataset into training (70%) and testing (30%) sets. Not all features were considered for classification, as many of them were mostly static or otherwise isolated in range. To demonstrate the early fitness of the algorithm for this use case, we used a subset of environmental sensors. The trained model produces the performance curves of Figures 3.8 and 3.9.

We can see that, in this case, the ROC curve is much simpler than the other use cases. The performance rates also show an inversion of the common behavior of the previous datasets: the false-positive rate is gradually dropping here, while the false-negative rate is abrupt. This is still a characteristic of the provided dataset, as abnormal readings have one or more features with error values that are atypical of that sensor's normal range, directly leading to a higher reconstruction error. Normal readings do not have a simple defining characteristic and their representations have more natural variance.

Here, a threshold value of $t = 0.3$ appears to correctly capture all the abnormal inputs of this test while keeping false positives relatively low, leading to an accuracy measure of 90%.

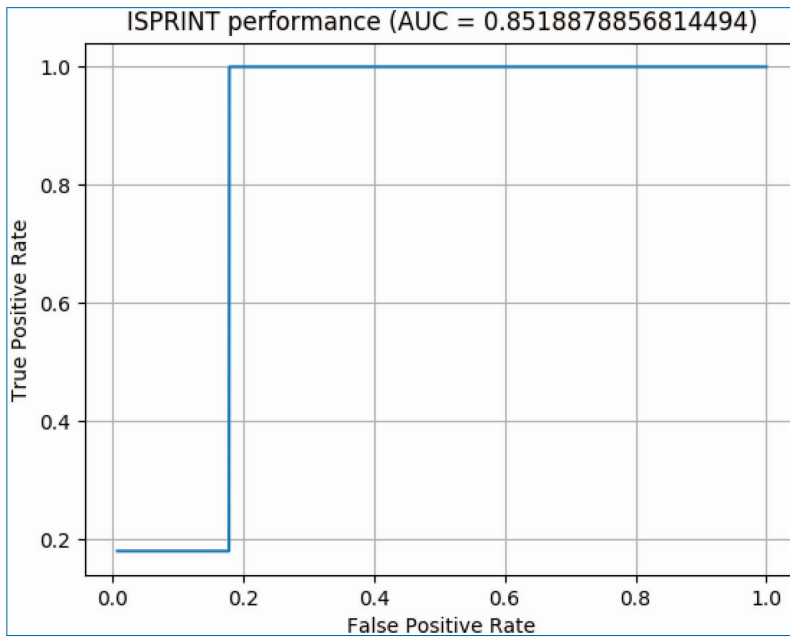


Figure 3.8. ROC curve of the CloudCare2U classifier.

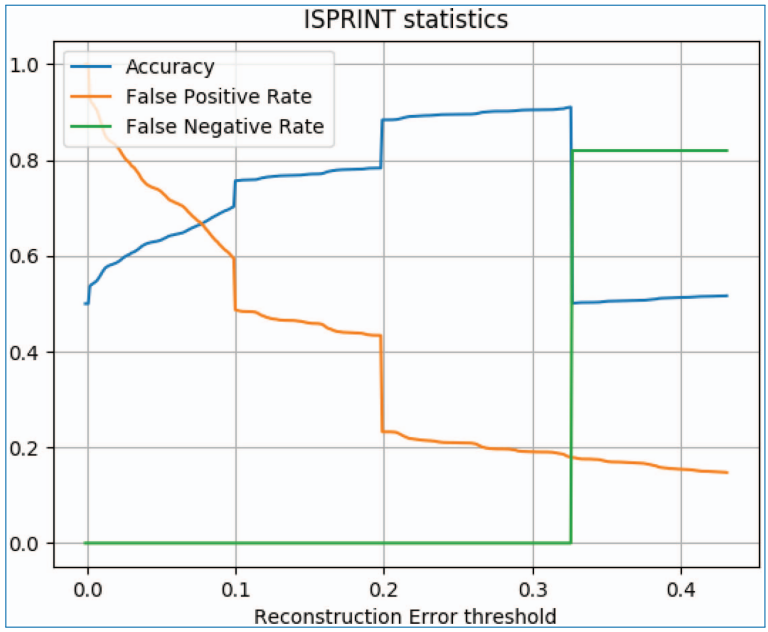


Figure 3.9. Performance variations on the CloudCare2U dataset as a function of the reconstruction error threshold.

3.4.3 Prototype Implementation

Apart from the training and evaluation of the VAE components on the above-listed datasets, we have also implemented and integrated them in the SecureIoT platform. Specifically, the Variational Autoencoder Analytics component has been developed into two levels. The first level is offering the algorithm itself with the offline training capabilities and the runtime version which discovers possible attacks. The second level is a "wrapper" to the Analytics algorithm which controls its lifecycle and generates compatible output for the SecureIoT infrastructure ready to be consumed by the Security-as-a-Service (SECaaS) services that are provided by the SecureIoT platform.

3.5 Conclusions and Discussion

This chapter has introduced a deep learning approach based on Variational Autoencoders for detecting anomalies in IoT applications that comprise smart objects like assistive robots and connected cars. The developed deep learning models managed to detect the abnormalities in the behavior of the IoT devices and smart objects, in three different application scenarios. Furthermore, the VAE components have been integrated in the SecureIoT platform for data-driven IoT security.

The developed model exhibits acceptable prediction time. Most of the time is spent on initializing the algorithm, i.e., loading the data and the model and compiling the model for the target platform. On an Intel Core-i7 4771 CPU and an nVidia Geforce GTX 770 GPU, initialization takes several seconds, while evaluation of the data currently takes 80 ns for the proposed model. In terms of scalability, the number of parameters of a model of a given depth scales linearly with the dimensionality of the input. As such, so do the training and evaluation times.

There is necessarily a stochastic element in training a deep learning model, which is further increased by using a variational approach. Nevertheless, a fully trained encoder will always output the same probabilistic parameters for the same input. Hence, the consistency of the model is deemed acceptable. Moreover, the current implementation requires a script that will invoke the model with the correct parameters and input files. However, the parameters do not change between invocations and the input file may or may not change, depending on the measurement storage process. In this way, some degree of automation can be achieved.

Overall, our work is one of the first efforts to exploit deep learning for detecting anomalies in smart objects. It is also an effort to integrate the deep learning model in an IoT security platform, i.e., to integrate deep learning with a real-life data collection system. However, future work is required towards identifying anomalies and assessing risks in more complex scenarios involving multiple smart objects and demanding correlation of diverse anomaly indicators. In this direction, relevant datasets need to be made available, beyond the publicly available network and IoT data.

Acknowledgments

This work has been carried out in the scope of the H2020 SecureIoT project, which is funded by the European Commission in the scope of its H2020 program (contract number 779899). The authors acknowledge valuable help and contributions from all partners of the project.

References

[1] Q. Jing, A. V. Vasilakos, J. Wan, J. Lu, and D. Qiu, "Security of the Internet of Things: perspectives and challenges," in Wireless Networks, vol. 20, issue 8, November 2014, pp. 2481–2501.

[2] A. Roukounaki, S. Efremidis, J. Soldatos, J. Neises, T. Walloschke, and N. Kefalakis, Scalable and Configurable End-to-End Collection and Analysis of IoT Security Data: Towards End-to-End Security in IoT Systems. GIoTS, 2019: 1–6.

[3] R. Martin, S. Schrecker, H. Soroush, J. Molina, J. P. LeBlanc, F. Hirsch, M. Buchheit, A. Ginter, H. Banavara, S. Eswarahally, K. Raman, A. King, Q. Zhang, P. MacKay, and B. Witten. (2016). Industrial Internet Security Framework Technical Report. 10.13140/RG.2.2.28143.23201.

[4] J. Soldatos, S. Kyriazakos *et al.* "Securing IoT Applications with Smart Objects: Framework and a Socially Assistive Robots Case Study," in Wireless Personal Communication Journal, to appear (2020), https://doi.org/10.1007/s11277-020-07039-1

[5] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, "Anomaly-based network intrusion detection: Techniques, systems and challenges," Comput. Secur., vol. 28, no. 1, pp. 18–28, 2009.

[6] S. X. Wu and W. Banzhaf, "The use of computational intelligence in intrusion detection systems: A review," Appl. Soft Comput., vol. 10, no. 1, pp. 1–35, 2010.

[7] J. Cannady, "Artificial neural networks for misuse detection," in Proc. 1998 Nat. Inf. Syst. Secur. Conf., Arlington, VA, USA, 1998, pp. 443–456.

[8] C. Livadas, R. Walsh, D. Lapsley, and W. Strayer, "Using machine learning techniques to identify botnet traffic," in Proc. 31st IEEE Conf. Local Comput. Netw., 2006, pp. 967–974.

[9] F. Jemili, M. Zaghdoud, and A. Ben, "A framework for an adaptive intrusion detection system using Bayesian network," in Proc. IEEE Intell. Secur. Informat., 2007, pp. 66–70.

[10] W. Li, "Using genetic algorithms for network intrusion detection," in Proc. U.S. Dept. Energy Cyber Secur. Group 2004 Train. Conf., 2004, pp. 1–8.

[11] M. Blowers and J. Williams, "Machine learning applied to cyber operations," in Network Science and Cybersecurity. New York, NY, USA: Springer, 2014, pp. 55–175.

[12] D. Bhamare, T. Salman, M. Samaka, A. Erbad, and R. Jain. (2016). Feasibility of Supervised Machine Learning for Cloud Security. 1–5. 10.1109/ICISSEC.2016.7885853.

[13] Z. Chkirbene, A. Erbad, and R. Hamila, "A Combined Decision for Secure Cloud Computing Based on Machine Learning and Past Information," 2019 IEEE Wireless Communications and Networking Conference (WCNC), Marrakesh, Morocco, 2019, pp. 1–6.

[14] D. Li and D. Yu, "Deep Learning Methods and Applications," in Foundations and Trends in Signal Processing, vol. 7, issue 3–4, 2014, pp. 197–387.

[15] M. Mohammadi, A. Al-Fuqaha, S. Sorour, and M. Guizani, "Deep Learning for IoT Big Data and Streaming Analytics: A Survey," IEEE Communications Surveys Tutorials, 2018.

[16] R. Lippmann *et al.*, "Evaluating intrusion detection systems: The 1998 DARPA offline intrusion detection evaluation," in Proc. IEEE DARPA Inf. Surviv. Conf. Expo., 2000, pp. 12–26.

[17] S. J. Stolfo, KDD Cup 1999 Data Set, University of California Irvine, KDD repository [Online]. Available: http://kdd.ics.uci.edu, accessed on Jun. 2014.

[18] M. Tavallaee, E. Bagheri, W. Lu, and A. Ghorbani, "A detailed analysis of the KDD Cup 1999 dataset," in Proc. 2nd IEEE Symp. Comput. Intell. Secur. Defense Appl., 2009, pp. 1–6.

[19] C. Shearer, "The CRISP-DM model: The new blueprint for data mining," J. Data Warehouse., vol. 5, pp. 13–22, 2000.

[20] C. G. E. Boender and A. R. Kan, "A Bayesian analysis of the number of cells of a multinomial distribution," The Statistician, pp. 240–248, 1983.

[21] [Online]. Available: https://commons.wikimedia.org/wiki/File:Perceptron.svg.

[22] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang, "On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima," 2017.

DOI: 10.1561/9781680836837.ch4

Chapter 4

Privacy Awareness, Risk Assessment, and Control Measures in IoT Platforms: BRAIN-IoT Approach

By Mohammad Rifat Ahmmad Rashid, Davide Conzon, Xu Tao and Enrico Ferrera

With the increasing adaptation of Internet of Things (IoT) platforms in decentralized cloud environments, more focus has been given towards facilitating the privacy awareness building upon goals set by current European Union (EU) General Data Protection Regulation (GDPR) regulations. Therefore, it is necessary to empower the end users (both private and corporate) of IoT platforms with the capability of deciding which combination of self-hosted or cloud-oriented IoT systems is most suitable to handle the personal data they generate and own as well as with the ability to change the existing (or pre-set) configurations at any time. BRAIN-IoT platform focuses on complex scenarios where actuation and control are cooperatively supported by populations of IoT systems. The breakthrough targeted by BRAIN-IoT is to provide solutions to embed privacy awareness and privacy control features in IoT solutions. In this work, the authors explore the following key areas: (a) privacy awareness in IoT systems using GDPR regulations and BRAIN-IoT platform, and (b) propose a conceptual framework for Privacy Impact Assessment (PIA) using privacy principles presented in GDPR regulations.

4.1 Introduction

In the recent years, increasing number of IoT products and services are being widely deployed in all professional and mass-market usage scenarios (Manyika *et al.*, 2015 and Gartner, 2015). This exponential growth of diverse IoT platforms leads to magnitude of market opportunities associated with the IoT products, but also in the rise of some associated challenges and concerns. Furthermore, the IoT technology and market landscape will become increasingly complex in the longer term, i.e., ten or more years from now, as indicated in Holland (2012), especially after IoT technologies will prove their full potential in business-critical and privacy-sensitive scenarios.

Privacy can be conceptualized as "the right to be left alone" (Panagiotou and Zaharakis, 2018). It refers to the process of disclosing and mobilizing personal data under certain conditions and safeguarding measures. From technical perspectives, IoT systems require context-based technological advancement regarding privacy awareness, keeping consumers convenience as the primary concern, as indicated by Sabo *et al.* (2013). Solutions suitable to tackle such challenges are still missing. For instance, in smart city scenarios, many initiatives fall into the temptation of developing new IoT platforms, protocols, models, or tools aiming to deliver the ultimate solution that will solve all the IoT challenges and become *the* reference IoT platform or standard. Instead, usually, they result in the creation of *yet-another* IoT solution or standard.

In this paper, the authors explore the challenges of privacy awareness in futuristic IoT environments, considering actuation in dependable fashion, by introducing privacy awareness and control approach in the BRAIN-IoT platform, which is a novel solution for building decentralized IoT platforms, based on inter-networking across heterogeneous existing IoT systems.

BRAIN-IoT has been designed to support dynamicity and heterogeneity in terms of new systems, protocols, and technologies, by allowing dynamic edge/-cloud reconfiguration. Another advanced approach proposed by BRAIN-IoT is security and privacy protection approach, which supports a set of end-to-end security, privacy, and trust enablers suitable for fully decentralized environments. In BRAIN-IoT platform, the authors address the challenges of privacy awareness for IoT platform using GDPR core principles. The authors divide this research goal into three research questions: (i) **RQ1**: Which features can be used to assess privacy awareness in the IoT platforms? (ii) **RQ2**: Which privacy impact assessment approach can be defined on top of the GDPR-based privacy principles? (iii) **RQ3**: How to validate the security and privacy impact of a given IoT platform?

In response to RQ1, the authors explore key privacy principles introduced in GDPR regulation and International Organization for Standardization

(ISO)/International Electrotechnical Commission (IEC) 29100 standard (Standardization, 2011). Using these privacy principles, the authors propose privacy assessment measures that can be used to detect privacy impact and address control criteria. To address RQ2, the authors propose an iterative PIA approach that profiles different privacy principles from GDPR and perform privacy compliance evaluation for the IoT platform. The main contributions of this work are: (a) privacy compliance evaluation approach using GDPR key privacy principles and ISO/IEC 29000 standard, (b) a conceptual framework for analyzing privacy risks using the measures from privacy compliance evaluation, and (c) a complete report about the experimentation of the proposed approach based on one of the BRAIN-IoT use cases (processed data with sensitive information), i.e., the Critical Water Infrastructure one, described in Table 4.1.

This paper is organized as follows: Section 4.2 presents the related work focusing on GDPR requirements related to IoT platforms, current standards, and tools for PIA. Section 4.3 describes the proposed conceptual framework that relies on the GDPR privacy principles and ISO/IEC 29100 standard. Section 4.4 outlines the privacy awareness and control integration approach applied in BRAIN-IoT Critical Water Infrastructure scenario, and Section 4.5 discusses the results obtained. Finally, the paper concludes with Section 12.5 by summarizing the main findings and outlining future research activities.

4.2 Literature Review

This section provides an overview of the State of The Art (SoTA) in the context of (i) GDPR implication in IoT domain and (ii) current standards and tools for PIA.

4.2.1 GDPR Requirements Related to IoT Domain

IoT applications are emerging across myriad sectors, for example, in health-care, energy consumption, and utility monitoring, as indicated by Wachter (2018). In these applications, to function properly as well as to optimize and customize their services, the IoT devices constantly collect vast amounts of personal data such as smart health applications that collect location data and health data (e.g., Fitbit) (Etsi, 2019). Without repeated and consistent identification of users, linked up, seamless services would not be possible. However, the pursuit of identification and personalization of users poses a risk to privacy.

Data controllers can draw inferences from these data. Users can easily perceive this insight as invasive, unexpected, and unwelcome. Discriminatory treatment can also result from inferential analytics and linkage of disparate records, motivating limitations on user profiling as stated by Wachter (2018). The impossibility of

weak cyber security standards, often owing to the limited computational power of identifying technologies such as Wireless Fidelity (WiFi) or Radio Frequency IDentification (RFID), can exacerbate privacy risks (Wachter, 2018). Together, these risks make free and well-informed consent challenging in the IoT. Considering the risk of personal data processing and linkage of user records, privacy policies often fail to communicate clearly the privacy risks, as indicated by Basso *et al.* (2015a).

In Libertés (2017), the authors explain that GDPR regulation creates governing principles of personal data processing (Articles 5) and guideline for individual rights (Article 25 and 33) that can help to perform PIA relevant for IoT devices. In particular, GDPR (Voigt and Bussche, 2017) helps for transparency (Article 5), data storage, access, rectification, and deletion (Articles 5, 15–17), informed consent (Article 7), notification duties (Articles 13–14 and 33), automated decision-making and profiling (Articles 21–22), privacy by design and privacy by default (Article 25), cyber security (Articles 33–34), and data protection impact assessment (Article 35–36). Tools created based on GDPR for PIA, like the one described in Libertés (2017), can help to undermine the tendency of IoT devices and services to collect, share, and store large and varied types of personal data, to operate seamlessly and covertly, and to personalize functions based on prior behavior. For example, EU H2020 project Synchronicity (Synchronicity, 2018) explores the benefit of GDPR privacy principles in the domain of smart city. In BRIAN-IoT context, the term privacy denotes "the right of an entity to be secure from unauthorized disclosure of personal information that are contained in BRIAN-IoT platform." In this paper, the authors explore the key privacy principles from GDPR to perform privacy compliance evaluation for any IoT system.

4.2.2 Current Standards and Tools for PIA

Personal information, in different forms, is continuously gathered and processed by modern applications, as indicated in Voigt and Bussche (2017). This includes data that are required for accessing a certain service (e.g., email address, credit card number), additional personal data that the user may provide for an enriched user experience (e.g., pictures, connections to social networks), and data that can be automatically gathered by the service provider (e.g., usage pattern, approximate location), as stated in Libertés (2017).

Standards for PIA. Taking into account various privacy principles, the fair information practices developed by the Organization for Economic Cooperation and Development (OECD) (OECD, 2013) and the Global Privacy Standard (Cavoukian, 2006) are commonly used in various application. However, less focus has been given towards privacy awareness and control in a decentralized environment. The most common reference models are the ISO/IEC 29100

(Standardization, 2011), the ISO/IEC 29101 (Standardization, 2013), the Organization for the Advancement of Structured Information Standards (OASIS) Privacy Management Reference Model (2013), and the reference architecture proposed by Basso *et al.* (2015b). The ISO/IEC 29100 describes a high-level framework for the protection of Personal Identifiable Information (PII), sets a common privacy terminology, defines privacy principles, and categorizes privacy features. Also, ISO/IEC 29100 is intended to be used by persons and organizations involved in designing, developing, procuring, testing, maintaining, and operating information and communication technology systems, where privacy controls are required for the functioning of PII.

This privacy framework is developed with the purpose of serving as assistance to organizations to define their privacy safeguarding requirements related to all information involved through following attributes: (i) by specifying a common privacy terminology, (ii) by defining the actors and their roles in processing PII, (iii) by describing privacy safeguarding considerations, and (iv) by providing references to known privacy principles for Information and Communication Technology (ICT). The ISO/IEC 29101 standard describes a reference architecture with best practices for a technical implementation of privacy requirements. It covers the various stages in data life cycle management and the required privacy functionalities for PII in each data life cycle, as well as positioning the roles and responsibilities of all involved parties in information and communication systems development and management. The OASIS Privacy Management Reference Model is a conceptual model, which helps understanding and implementing appropriate operational privacy management functionalities and supporting mechanisms capable of executing privacy controls in line with privacy policies. These standards can be used to guide, design, develop, and implement privacy policies and controls, they can also be used as a reference point in the monitoring and measurement of performance benchmarking and auditing aspects of privacy management programs in an organization. In the BRAIN-IoT context, apart from GDPR regulations, the Consortium explores ISO/IEC 29100 standard to identify privacy control criteria.

Tools for PIA. The Unified Modeling Language (UML) (Group, 2011) is a central resource in the development of modern software systems. The objective of UML is to provide software engineers with tools for analysis, design, and implementation of software-based systems, as well as for modeling other kind of processes. The UML specification defines a lightweight mechanism for extending the language, called "profiling." In Basso *et al.* (2015a), the authors propose a UML profile for privacy-aware applications. This UML-based profiling approach helps building UML models that specify and structure concepts of privacy. Although UML-based profiling can help to model the privacy criteria, less focus has been given towards generalizing privacy control approach for IoT platforms.

Considering tools designed for privacy awareness and control, the nationale de l'informatique et des libertés (CNIL) PIA tool described in Libertés (2017) has been designed around three principles: (i) a didactic interface to carry out PIAs: the tool relies on a user-friendly interface to allow for a simple management of your PIAs. It clearly unfolds the PIA methodology step by step. Several visualization tools offer ways to quickly understand the risks. (ii) A legal and technical knowledge base: the tool includes the legal points ensuring the lawfulness of processing and the rights of the data subjects. (iii) A contextual knowledge base, available along all the steps of the PIA, adapting the contents displayed. The data are extracted from several sources, e.g., the GDPR, the PIA guides and the Security Guide from the CNIL, to the aspect of the processing studied. BRAIN-IoT will extend the IoT-Modeling Language (IoT-ML) adding the some specific modeling profile that can be used to describe privacy awareness polices. IoT-ML is a UML-based modeling language built upon the Internet of Things Architecture (IoT-A) architecture framework, described in Atzori (2017). It adds to the key concepts provided by IoT-A to describe IoT systems, the ability to integrate new IoT domain models, as indicated in Gérard *et al.* (2017).

4.3 A Conceptual Privacy Awareness Framework

A PIA approach can assist the controller of an IoT system to identify any specific risks of privacy breaches involved in an envisaged operation. More specifically, a PIA is an essential approach to identify privacy risks, and perform risk analysis, risk evaluation, personal data flow, and storage. In the BRAIN-IoT scope, the PIA can be performed in the context of two components, IoT devices and IoT systems for various use cases.

The proposed approach is based on the following four components: (i) *Context*: Define and describe the context of the IoT system under consideration. (ii) *Privacy principles*: Analyze the controls guaranteeing compliance with the GDPR fundamental principles considering the proportionality and necessity of processing, and the protection of data subjects rights. (iii) *Privacy risks*: Identify the threats that are associated with data privacy and ensure they are properly treated. (iv) *Privacy compliance evaluation*: Analyze the overall privacy safeguarding requirements related to the processing of information assets in a particular setting.

4.3.1 Context

At the beginning of a PIA, it is important to outline the use case under consideration, its nature, scope, context, and purposes. Also, identify the data controller and any processor of information assets in a particular setting. In the context of

BRAIN-IoT, for PIA an information asset could be: (a) software (e.g., an operating system), (b) hardware (e.g., a sensor, Central Processing Unit – CPU – memory, etc.), (c) processed data (e.g., personal data stored in BRAIN-IoT platform, sensor status transmitted over a network, robot location in memory, etc.). In certain instances, identifiability of the processed data might vary. To determine whether or not information about a person considered as processed data in IoT platform, several factors need to be taken into account. Personal data can be considered to be information assets in at least the following instances (Standardization, 2011): (i) if it contains or is associated with an identifier, which refers to a natural person; (ii) if it contains or is associated with an identifier, which can be related to a natural person; (iii) if it contains or is associated with an identifier, which can be used to establish a communication with an identified natural person, and (iv) if it contains references, which link the data to any of the identifiers above. Information does not need to have an identifier to be considered information asset. Information will also be considered personal data, if it contains or is associated with a characteristic which distinguishes specific purpose. Any attribute which takes on a value that uniquely identifies a information assets has to be considered as a distinguishing characteristic, as indicated in Standardization (2011). Further, the authors explore the processed data based on three categories: (a) information about the user (first name, date of birth, email, etc.), (b) recorded data (sounds, images, movements, temperature, etc.), and (c) calculated data (data from robot operating system, etc.). These concepts have been used to define a template to provide a brief description of the use cases for PIA.

4.3.2 Privacy Principles

The key element in a PIA is the evaluation of how an IoT platform provide services for collect, store, and distribute personal data, as described in Basso *et al.* (2015b). In the GDPR regulation (Parliament, 2016), the six key principles are set out right at the beginning and inform everything that follows. Compliance with these key principles is, therefore, a fundamental building block for PIA. In this PIA, apart from six key principles, which lie at the heart of the GDPR, four additional privacy principles are considered: *(a) the right to be informed*, *(b) the rights of access and to data portability*, *(c) the rights to rectification and erasure*, and *(d) the rights to the restriction of the process and to object*. These four principles are based on the regulation for individuals' rights presented in GDPR (Article 15–17, 20, 25 and 33). These four additional principles are considered based on the premise that individuals' rights for access, rectification, erasure, portability and notifications for personal data are important aspects for an IoT platform, as indicated in Libertés (2017) and

Basso *et al.* (2015b). This section outlines privacy control criteria based on the six key principles and four additional principles.

(P1) Lawfulness, fairness, and transparency: The lawfulness, fairness, and transparency for information assets are set out in Article 6 of the GDPR. Six privacy control criteria are defined for this principle (P1-01...P1-06, corresponding to GDPR Article 6 points a...f). At least one of these criteria must apply whenever processing personal data.

(P2) Purpose limitation: In the BRAIN-IoT context, purpose limitation principle helps to ensure controller plan to use or disclose processed data (GDPR article 5). The controller needs to specify purpose that is additional or different from the original specified purpose. Also, it is important that the new specification must be fair, lawful, and transparent (Standardization, 2011). In this case, the criteria are: P2-01: clearly identified purposes for data processing; P2-02: purposes must be documented; P2-03: regularly review the processing and, where necessary, update the documentation and privacy information for individuals.

(P3) Data minimization: Data minimization is closely linked to the principle of "collection limitation" but goes further than that. Whereas "collection limitation" refers to limited data being collected in relation to the specified purpose, "data minimization" strictly minimizes the processing of information assets (GDPR article 5). The criteria for this principle are: P3-01: ensure adoption of a "need-to-know" principle, i.e., one should be given access only to the information assets which is necessary for the conduct of his/her official duties; P3-02: delete and dispose of information asset whenever the purpose for personal data processing has expired; there are no legal requirements to keep the information asset or whenever it is practical to do so.

(P4) Data quality: The data accuracy controls criteria for BRAIN-IoT system requirements based on the Article 5(1)(d) of the GDPR (Parliament, 2016) are: P4-01: ensure the processed data present in the IoT platform is not incorrect or misleading; P4-02: ensure the reliability of processed data collected from a source; P4-03: keep a note of any challenges to the accuracy of the IoT platform processed data.

(P5) Storage limitation: Article 5(1)(e) of GDPR explicitly pointed out that a storage duration must be defined for each type of process data, and it must be justified by the legal requirements and processing needs (Parliament, 2016). Thus, storage of personal data in a IoT platform must be justified using privacy control criteria. The evaluation criteria for the storage limitation in the context of BRAIN-IoT system are: P5-01: ensure mechanism must be implemented to archive common data or purge archived data at the end of their storage duration; P5-02: functional

traces will also have to be purged, as will technical logs which may not be stored indefinitely.

(P6) Integrity and confidentiality: Article 5(1)(f) of the GDPR concerns the "integrity and confidentiality" of personal data. It can be referred to as the GDPR's "security principle." This principle explores the broad concept of information security. Poor information security leaves any systems and services at risk and may cause real harm and distress to individuals—lives may even be endangered in some extreme cases. The common control criteria for integrity and confidentiality are: P6-01: presentation, when the device is activated, of the terms and conditions for use/confidentiality; P6-02: possibility of accessing the terms & conditions for use/confidentiality after activation; P6-03: legible and easy-to-understand terms; P6-04: existence of clauses specific to the device; P6-05: detailed presentation of the data processing purposes (specified objectives, data matching where applicable, etc.); P6-06: detailed presentation of the personal data collected; P6-07: presentation of any access to the identifiers of the device, the smartphone/tablet or computer, specifying whether these identifiers are communicated to third parties; P6-08: information for the user if the app is likely to run in the background; P6-09: information on the secure data storage method, particularly in the event of sourcing; P6-10: information on protection of access to the device; P6-11: arrangements for contacting the company (identity and contact details) about confidentiality issues; P6-12: where applicable, information for the user on any change concerning the data collected, the purposes, and confidentiality clauses.

(P7) The right to be informed: In the BRAIN-IoT platform, various components of IoT systems are integrated for modeling tasks. An IoT platform should be able to demonstrate that an individual has consented the processed data. The individual must be able to withdraw his/her consent easily at any time (Articles 7 and 8 of the GDPR). The following is a list of controls intended to ensure that: users of BRAIN-IoT consent has been informed; there has been a reminder and confirmation of their consent; the settings associated with the latter have been maintained: P7-01: express consent during activation; P7-02: consent segmented per data category or processing type; P7-03: express consent prior to sharing data with other users; P7-04: consent presented in an intelligible and easily accessible form, using clear and plain language adapted to the target user; P7-05: for each new user, consent must once again be obtained; P7-06: where the user has consented to the processing of special data (e.g., his/her location), the interface clearly indicates that said processing takes place (icon, light).

(P8) The rights of access and to data portability: Where the data processing benefits from an exemption from the right of access, as presented in Articles 15 of

the GDPR, it is necessary to describe their implementation on the IoT platform, as well as a justification on the arrangements. The following is a list of the controls intended to ensure user's rights of access to all data concerning restriction in access and portability: P8-01: possibility of accessing all data, via the common interfaces; P8-02: possibility of securely consulting the traces of use associated with the user; P8-03: possibility of downloading an archive of all the processed data associated with the use; P8-04: possibility of retrieving, in an easily reusable format, personal data provided by the user, to transfer them to another service (Article 20 of the GDPR)

(P9) The rights to rectification and erasure: As presented in Article 17 of the GDPR, it is required to justify the arrangement for responding to the individuals considering the rights to rectification and erasure. This is the list of controls intended to ensure the right of rectification and erasure of the processed data: P9-01: indication from the IoT platform that the processed data will nevertheless be stored (technical requirements, legal obligation, etc.); P9-02: clear indications and simple steps for erasing data before scrapping the device; P9-03: possibility of erasing the data in the event the device is stolen.

(P10) The rights to restriction of processing and to object: The list of control criteria considering the rights to restriction of processing and to object of personal data (from GDPR article 20 and 25) in a IoT platform are: P10-01: existence of "Privacy" settings; P10-02: invitation to change the default settings; P10-03: "Privacy" settings accessible when activating the device; P10-04: "Privacy" settings accessible after activating the device; P10-05: existence of technical means for the data controller to lock access to and use of the individuals to restriction; P10-06: possibility of deactivating some of the device's features (microphone, Web browser, etc.); P10-07: existence of alternative apps for accessing the device; P10-08: compliance in terms of tracking; P10-09: effective exclusion of processing the user's data in the event consent is withdrawn.

4.3.3 Privacy Risks

GDPR is a complex regulation, but at its core, it's a data protection law. This means a processor must explore privacy threats and vulnerabilities to personal data protection to assess and mitigate privacy risks under GDPR. According to ISO/IEC 29100 standard, privacy threat is effect of uncertainty on privacy. Based on the threats and vulnerabilities explored in various studies, e.g., in Wachter (2018), Parliament (Parliament, 2016), and Libertés (2017), the list of privacy threats in the context of BRAIN-IoT platform can be found in BRAIN-IoT (2018), results of the threat modeling done by the partners of the project.

4.3.4 Privacy Compliance Evaluation

In this approach, the privacy compliance evaluation is performed using the fundamental privacy principles presented in GDPR and privacy threats outlined in Section 4.3.3. The privacy compliance evaluation is based on two pillars: *(1) Fundamental rights and principles*, which are "non-negotiable," established by law, and which must be respected, regardless of the nature, severity, and likelihood of risks; *(2) Privacy threats*, which identify the appropriate technical and organizational risks to protect individuals data.

In any IoT application, the controller of a use case should identify and implement privacy controls to meet the privacy safeguarding requirements, which is identified by the privacy risk assessment and treatment process, as indicated in Libertés (2017). Effort should be taken by IoT application controllers to develop their privacy controls as part of a general "privacy by design" approach considerations, i.e., privacy compliance evaluation should be considered at the design phase of systems data processing, rather than being bolted on at a subsequent stage, as suggested in Voigt and Bussche (2017). Therefore, it is important to verify and demonstrate that the processing meets data protection and privacy safeguarding requirements by periodically conducting evaluation using internal auditors or trusted third-party auditors, as stated in Libertés (2017). Also, the identified and implemented privacy controls should be documented as part of the PIA for IoT applications.

Certain types of data processing can warrant specific controls for which the need only becomes apparent once an envisaged operation has been carefully analyzed. A risk is a hypothetical scenario that describes a feared event and all the threats that would allow this to occur (Article 32 of the GDPR). Based on the privacy risk level outlined in Figure 4.1, the authors performed the assessment of privacy control criteria using the following scale to estimate the Severity and likelihood of privacy risks:

Negligible (Scale: 0): It does not seem possible for the selected risk sources to materialize the threat by exploiting the properties of supporting assets (e.g., theft of paper documents stored in a room protected by a badge reader and access code). Considering severity, data subjects either will not be affected or may encounter a few inconveniences, which they will overcome without any problem.

Limited (Scale: 1): It seems difficult for the selected risk sources to materialize the threat by exploiting the properties of supporting assets (e.g., theft of paper documents stored in a room protected by a badge reader). Privacy stakeholders may encounter significant inconveniences, which they will be able to overcome despite a few difficulties.

Significant (Scale: 2): It seems possible for the selected risk sources to materialize the threat by exploiting the properties of supporting assets (e.g., theft of paper

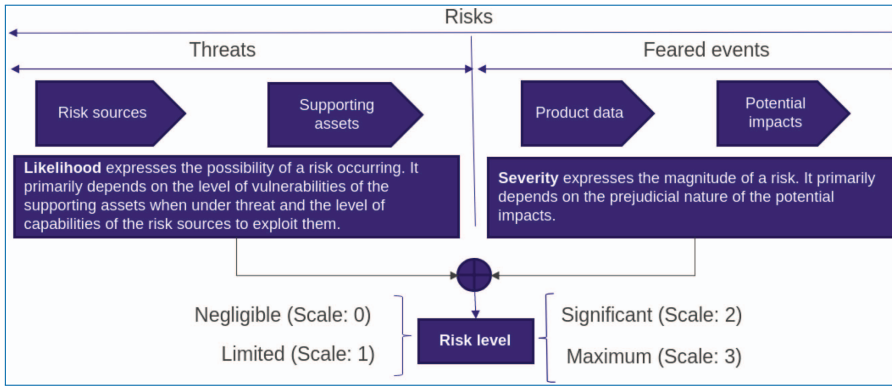

Risks
Threats
Feared events
Risk sources
Supporting assets
Product data
Potential impacts
Likelihood expresses the possibility of a risk occurring. It primarily depends on the level of vulnerabilities of the supporting assets when under threat and the level of capabilities of the risk sources to exploit them.
Severity expresses the magnitude of a risk. It primarily depends on the prejudicial nature of the potential impacts.
Negligible (Scale: 0)
Limited (Scale: 1)
Risk level
Significant (Scale: 2)
Maximum (Scale: 3)

Figure 4.1. Privacy and security risk level.

documents stored in offices that cannot be accessed without first checking in at the reception). Privacy stakeholders may encounter significant consequences, which they should be able to overcome albeit with real and serious difficulties.

Maximum (Scale: 3): It seems extremely easy for the selected risk sources to materialize the threat by exploiting the properties of supporting assets (e.g., theft of paper documents stored in the public lobby). Further, privacy stakeholders may encounter significant, or even irreversible consequences, which they may not overcome.

Considering the likelihood and severity scale, it is possible to evaluate a privacy principle (P) using each privacy control criterion (C_i). The risk level for a privacy principle (P) can be defined as:

$$P = \frac{1}{n} \sum_{i=1}^{n} C_i, i = 1 \ldots n \tag{4.1}$$

Here, n is the number of privacy control criteria. Based on the risk value of a privacy principle, it can be assumed that if the value is greater than 2, then the risk level for a use case is significant and need further attention from controllers. The severity and likelihood evaluation can be performed by monitoring how many information assets affected by a specific privacy threat.

4.4 Experimental Analysis

This section describes the experiments performed on IoT scenario, namely Critical Water Infrastructure, provided by the BRAIN-IoT project. Firstly, an use-case

description is presented, and then, the results of privacy compliance analysis are reported.

Below in Table 4.1, the authors present an overview of Critical Water Infrastructure use case. Considering the processed data, there are no real personal data present in this use case. However, the generalized PIA approach based on GDPR, presented in Section 4.3, is applicable also to this type of use cases. Based on this approach, the authors can get an overall view on impact of privacy threats considering GDPR regulation. The result of the privacy compliance evaluation of Critical Water Infrastructure scenario is briefly presented in Table 4.2.

Table 4.1. Critical infrastructure management.

Use-case description	The services of water treatment and supply in large cities are fundamental for the existence of life and for its quality. These services are associated to several infrastructures that are considered, in accordance with European norms, as critical, and as such are therefore bound to a series of conditions for their development. Within this context, it's about how best to apply the technological improvements to our work environment preserving the established guarantees. It's about progressing in the application of technology in the processes related to potable water and its urban distribution, in a way that within the sustainability context, citizens receive a better service. In other words, from a social point of view, but also from economic environmental one.
Processing purposes	An important part of this effort is orientated to systems' implementation, improving the general information of the companies processes, thus those associated with IoT technologies and their integration as company's information models. The idea is that a greater amount and better flow of information will allow taking better decisions, to achieve resilient and sustainable systems to guarantee high quality of life.
Processing stakes	Online monitoring system of the quality of surface waters, with two profilers and two quality control stations in the dam and the river that supply the metropolitan area around the city of Coruña. Treatment Plant (ETAP), with an automatized management system that allows to control all process aspects to optimize the inputs demand, especially reactive and energy as fundamental elements of sustainability. Tele-control system of supply grids, with more than 80 control points, i.e., deposits, pumps, and sectors, with access to water quality data and the correct infrastructures' functioning. This system serves also as a feed of the hydraulic model of the supply grid, allowing a constant improvement of water distribution efficiency amongst the population.

(*Continued*)

Table 4.1. Critical infrastructure management (continued).

	Monitoring of 20 control sections of division consumption with more than 2000 metres installed to profile consumption patterns and understand the behavior of the consumption for its implementation in hydraulic and commercial models to improve the short-term services. Monitoring of big clients' systems with multi-parameter stations. Monitoring of the delivery of water to third parties.
Processed data	*(a) Information about user*: No personal data is stored for processing.
	(b) Recorded data: (1) SERVER IP (Station A TELVA), (2) SERVER 01 (Win CC System), (3) SERVER 03 (Station and Central Server), (4) Meteorological device, (5) Industrial Ethernet Switch not managed, (6) PC1-SCADA WinCC NT, (7) PC2-SCADA EMALCSA, (8) Router, (9) Gaugin Station, (10) Flow metre Sitrans MAG 5000, (11) Flow metre Sitrans MAG 8000, (12) Flow metre Fus-1010, (13) Flow metre Woltman, (14) Flow metre Vortex.
	(c) Calculated data: (1) Data from dam level probe, (2) Multi-parameter probe for water quality monitoring, (3) Secondary hydrostatic dam level probe, (4) PLC module in dam top, (5) PLC in Dam Room, (5) Interface module, (6) Profinet, (7) Profibus, (8) SensiNact, (9) Tacdesoft new remote station, and (10) Profibus Repeater
Controller	Infrastructure Management (EMALCSA)
Processor(s)	BRAIN-IoT platform

Table 4.2. Privacy compliance evaluation of critical infrastructure use case.

		Risk level	
Privacy control criteria	**Privacy threats**	**Severity scale**	**Likelihood scale**
(P1) Lawfulness, fairness, and transparency			
P1-01	R-07, R18, R21	2	1
P1-02	R-19, R-20	1	0
Other rows are omitted for brevity			
P1-06	R-18, R-19, R-21	3	2
(P2) Purpose limitation			
P2-01	R-16	1	1
P2-02	R-06, R-08	2	0
P2-03	R-08, R-20	0	1
(P3) Data minimization			
P3-01	R-01, R-04, R-06	3	2
P3-02	R-16	2	2

(*Continued*)

Table 4.2. Privacy compliance evaluation of critical infrastructure use case (continued).

Privacy control criteria	Privacy threats	Risk level	
		Severity scale	Likelihood scale
	(P4) Data quality		
P4-01	R-01	2	1
P4-02	R-08, R-15	0	1
P4-03	R-11	1	0
	(P5) Storage limitation		
P5-01	R-06, R08, R15	1	2
P5-02	R-15, R-18	0	1
	(P6) Integrity and confidentiality (Security)		
P6-01	R-10, R14	2	1
P6-02	R-07	1	3
	Other rows are omitted for brevity		
P6-12	R-16, R-18, R-20	2	2
	(P7) The right to be informed		
P7-01	R-06, R-07	2	1
P7-02	R-05, R-07	2	3
	Other rows are omitted for brevity		
P7-06	R-02, R-11	2	1
	(P8) The rights of access and to data portability		
P8-01	R-01	2	3
P8-02	R-02	1	2
P8-03	R-07	1	0
P8-04	R-19	2	2
	(P9) The rights to rectification and erasure		
P9-01	R-12	1	0
P9-02	R-05,R-06,R-09	2	2
P9-03	R-05	3	2
	(P10) The rights to restriction of processing and to object		
P10-01	R-07	1	2
P10-02	R-10	1	0
	Other rows are omitted for brevity		
P10-09	R-19	1	2

4.5 Discussion

This section reports a synthesis of the main findings of the experimental analysis presented in Section 4.4.

The privacy compliance analysis is presented in a radar chart in Figure 4.2. This figure illustrates the PIA results in a radar chart using 10 (P1 to P10) GDPR-based privacy principles. The severity (expresses the magnitude of a risk) and likelihood (expresses the possibility of a risk occurring) evaluation can be performed by monitoring how many information assets (such as door controller, etc.) are affected by a specific privacy threat (for instance, unauthorized access to services, illegal access, etc.). The scale used is the one presented in Section 4.3, so it can be assumed that if the risk value is greater than 2, then the risk for a use case is significant and needs further attention from the use-case owner. From the PIA results, it is possible to see that Integrity and confidentiality (Security) privacy principle (P6) has high severity and likelihood scale value (2) due to risks regarding illegal access, unauthorized access to services, data from untrustworthy sources, vulnerable IoT devices, and data from untrustworthy sources. Figure 4.3 illustrates the main findings of this work which is labeled A to F.

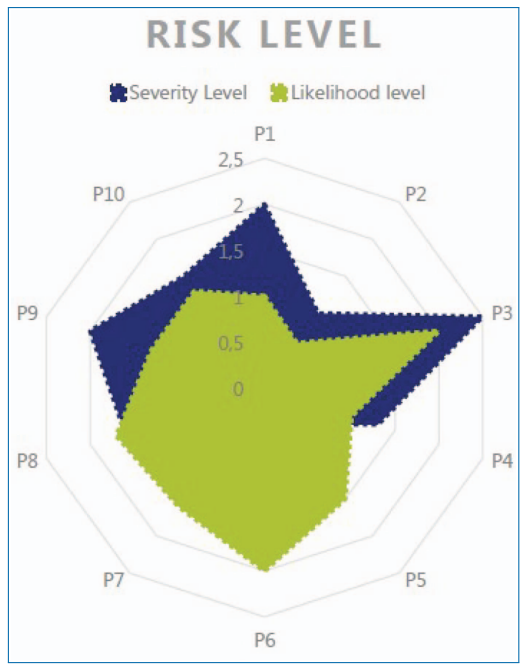


Figure 4.2. Privacy compliance analysis for critical infrastructure scenario.

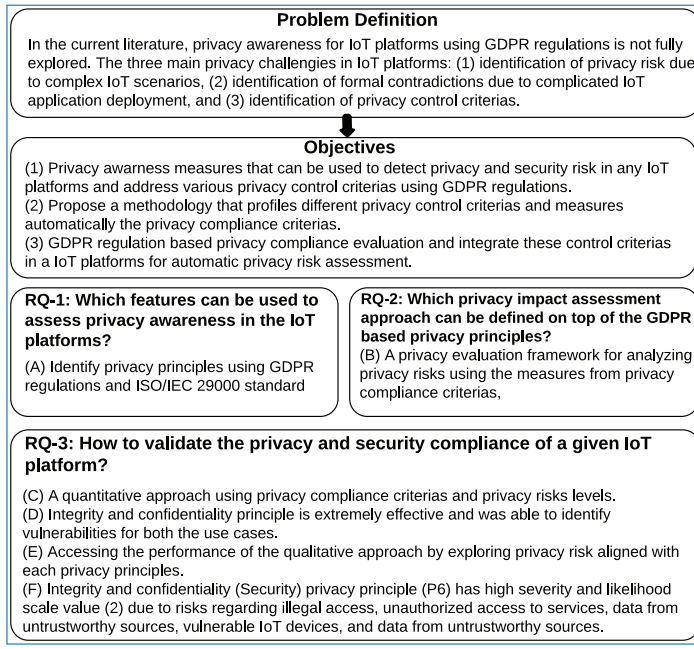

Problem Definition
In the current literature, privacy awareness for IoT platforms using GDPR regulations is not fully explored. The three main privacy challengies in IoT platforms: (1) identification of privacy risk due to complex IoT scenarios, (2) identification of formal contradictions due to complicated IoT application deployment, and (3) identification of privacy control criterias.
Objectives
(1) Privacy awarness measures that can be used to detect privacy and security risk in any IoT platforms and address various privacy control criterias using GDPR regulations.
(2) Propose a methodology that profiles different privacy control criterias and measures automatically the privacy compliance criterias.
(3) GDPR regulation based privacy compliance evaluation and integrate these control criterias in a IoT platforms for automatic privacy risk assessment.
RQ-1: Which features can be used to assess privacy awareness in the IoT platforms?
(A) Identify privacy principles using GDPR regulations and ISO/IEC 29000 standard
RQ-2: Which privacy impact assessment approach can be defined on top of the GDPR based privacy principles?
(B) A privacy evaluation framework for analyzing privacy risks using the measures from privacy compliance criterias,
RQ-3: How to validate the privacy and security compliance of a given IoT platform?
(C) A quantitative approach using privacy compliance criterias and privacy risks levels.
(D) Integrity and confidentiality principle is extremely effective and was able to identify vulnerabilities for both the use cases.
(E) Accessing the performance of the qualitative approach by exploring privacy risk aligned with each privacy principles.
(F) Integrity and confidentiality (Security) privacy principle (P6) has high severity and likelihood scale value (2) due to risks regarding illegal access, unauthorized access to services, data from untrustworthy sources, vulnerable IoT devices, and data from untrustworthy sources.

Figure 4.3. Summary of findings.

4.6 Conclusion and Future Work

Complex IoT application scenarios present a set of challenges that were not considered in more traditional IoT applications, such as heterogeneity and lack of interoperability, security and privacy concerns. Moreover, the challenges in implementing privacy awareness in IoT platforms consider the needs of allowing autonomous behaviors in isolation. Further, as more IoT systems get connected to the internet, there is a need for a highly scalable and extensible PIA approaches. Authors' belief is that adopting the GDPR key privacy principles to remodel PIA approach will enhance the reliability of IoT platform. This paper has explored the lingering problems in the context of privacy awareness in complex IoT applications by integrating GDPR in IoT platforms. More specifically, this paper has presented the way in which the BRAIN-IoT project intends to address challenges of privacy awareness in IoT platforms by leveraging GDPR and ISO/IEC standards. This paper has further proposed a PIA approach for IoT platforms, which will also be advantageous from the developers' point of view. The proposed framework is yet to be fully implemented and tested but it presents a direction of investigation for the IoT community on ways the problem of scalability, ease of development, ease of deployment, and lightweight implementation can be resolved.

Acknowledgments

This work is supported by European Union's Horizon 2020 research and innovation programme under grant agreement No 780089, project BRAIN-IoT (model-Based fRamework for dependable sensing and Actuation in INtelligent decentralized IoT systems).

References

Atzori, L., A. Iera, and G. Morabito. 2017. "Understanding the Internet of Things: definition, potentials, and societal role of a fast evolving paradigm." *Ad Hoc Networks*. 56: 122–140.

Basso, T., M. Leonardo, M. Regina, and J. M. e B. Andrea. 2015a. "Towards a UML profile for privacy-aware applications." In: *Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence*. Ed. by I. International. and Computing.

Basso, T., R. Moraes, and M. J. e M. Vieira. 2015b. "Requirements, design and evaluation of a privacy reference architecture for web applications and services." In: *Proceedings of the 30th Annual ACM Symposium on Applied Computing*.

BRAIN-IoT, C. 2018. "Initial Threat modelling and Security assessment." *Tech. Rep.*

Cavoukian, A. 2006. *Creation of a Global Privacy Standard*. [Online]. Available. URL: http://www.ehcca.com/presentations/privacysymposium1/cavoukian%5C_2b%5C_h5.pdf.

Etsi, C. Y. B. E. R. 2019. "Cyber Security for Consumer Internet of Things." *ETSI*. 2019.

Gartner. 2015. "M2M Global Forecast & Analysis Report." *Tech. Rep.* Machine Research.

Gérard, S., C. Dumoulin, P. Tessierck, and B. Selic. 2017. "Papyrus: A UML2 Tool for Domain-Specific Language Modeling." pp. 361–368.

Group, O. M. 2011. "OMG Unified Modeling Language (OMG UML)." *Superstructure, Version*. 2(4): 1.

Holland, J. H. 2012. *Signals and Boundaries: Building Blocks for Complex Adaptive Systems*. MIT Press.

Libertés, C. N. de l'Informatique des. 2017. *Privacy Impact Assessment (PIA)*. [Online]. Available. URL: https://www.cnil.fr/en/privacy-impact-assessment-pia.

Manyika, J., M. Chui, P. Bisson, J. Woetzel, R. Dobbs, J. Bughin, and D. Aharon. 2015. *Unlocking the Potential of the Internet of Things*. McKinsey Global Institute.

OECD. 2013. "OECD guidelines governing the protection of privacy and transborder flows of personal data." *Tech. Rep.*

Panagiotou, P. and N. S. e I. D. Zaharakis. 2018. "Design and Implementation of a Privacy Framework, for the Internet of Things (IoT)." In: *21th EU-ROMICRO Conference on Digital System Design.* Methods, Tools (DSD'18), Prague, Czech Republic: Architectures.

Parliament, E. 2016. "Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46." *Official Journal of the European Union (OJ)*. 59: 1–88.

Sabo, J., M. Willett, and P. B. e D. N. Jutla. 2013. *Privacy Management Reference Model and Methodology*. OASIS PMRM TC Standards Track Committee Specification.

Standardization, I. O. 2011. *ISO/IEC 29100.International Standard for Information technology – Security Techniques and Privacy Framework*. First. Privacy framework.

Standardization, I. O. 2013. *ISO/IEC 29101. International Standard – Information Technology – Security Techniques – Privacy Architecture Framework*. First.

Synchronicity, C. 2018. "D2.10 Reference Architecture for IoT Enabled Smart Cities, Update." *Tech. Rep.*

Voigt, P. and A. V. dem Bussche. 2017. *The EU general data protection regulation (GDPR)*. 1st. A Practical Guide, Cham: Springer International Publishing.

Wachter, S. 2018. "Normative challenges of identification in the Internet of Things: Privacy, profiling, discrimination, and the GDPR." *Computer Law & Security Review*. 34(3): 436–449.

DOI: 10.1561/9781680836837.ch5

Chapter 5

IoT Network Risk Assessment and Mitigation: The SerIoT Approach

By Gianmarco Baldini, Piotr Fröhlich, Erol Gelenbe, Jose Luis Hernandez-Ramos, Mateusz Nowak, Slawek Nowak, Stavros Papadopoulos, Anastasis Drosou and Dimitrios Tzovaras

Cyberattacks on the Internet of Things (IoT) can be the source of major economic damage. They can disrupt production lines, manufacturing processes, and supply chains. They can adversely impact the physical safety of vehicles and transportation systems, and damage the health of living beings both through supply chains for food, medicines, and other vital items, as well as through direct attacks on sensors and actuators that may be connected to vital functions. Thus, securing the IoT is of primary importance to our societies. This paper describes the technical approach that we adopt for IoT security in the SerIoT Research and Innovation Project that is funded by the European Commission. We first discuss the risk scenario for the IoT and briefly review approaches that have been developed to mitigate such risks. Then, we discuss a policy-based lightweight approach that mitigates risks at the level of the attachment of IoT devices to a network. We follow this with a detailed proposal based on using a distributed Machine Learning approach to risk and attack detection in real time, as well as suggestions for future work.

5.1 Introduction

The IoT may extend to billions of objects and sensors connected to Clouds, databases, decision systems, and actuators [1]. It has the potential to improve the critical processes that are at the heart of our socio-economic systems [2, 3] composing a data-driven society [4]. However, the pervasive nature of the IoT raises risks that go way beyond the individual technologies such as the internet or wireless networks [5] and machine-to-machine systems [6]. In addition to risks related to system malfunctions, Quality of Service (QoS) failures, and excessive energy consumption, they also include the theft and tampering of data, conventional network attacks, and other attacks that attempt to deplete the energy of autonomous sensors and actuators [7–12].

In Information and Communication Technology (ICT), risk management embraces different processes intended to deal with the identification, assessment, and mitigation of risks, which are derived from vulnerable ICT systems or potential cybersecurity attacks [13]. However, the scale and heterogeneity of IoT systems sets out significant challenges for the implementation of a risk management approach. On the one hand, IoT represents the current trend to hyperconnected systems; therefore, risk management aspects must consider the relationships and dependencies among different devices that could affect to the security level of a certain system or deployment. On the other hand, IoT deployments are usually composed by components with resource constraints, which are installed in uncontrolled environments with default security configurations. These constraints also make IoT devices and systems an attractive target for possible attacks. Furthermore, due to the potential huge number of devices in IoT deployments, there is a real need to consider risk management approaches with a high degree of automation to efficiently identify and mitigate new risks affecting such systems. These approaches should be able to represent the relationships among devices, vulnerabilities, and attacks of the overall deployment.

To create a suitable risk management methodology for the IoT, holistic approaches are required to understand the probability and impact of potential risks affecting devices and systems. As a core process of risk management, risk assessment has been strongly considered in recent years by IoT researchers [14]. However, assessing risk is in itself insufficient, and we must design IoT networks that can detect and then mitigate security risks, but also mitigate other risks that may arise regarding the overall performance of the system by preserving QoS and offering energy efficient operation of the system [15]. Thus, a continuing overall evaluation of the risks introduced by IoT systems and their networks is needed, and this paper offers a view of risk identification and mitigation as applied to IoT systems, based on our work in the SerIoT Project supported by the

European Commission [16]. In particular, we propose a system called *Autopolicy*, which is intended to enforce security profiles according to the intended communications of a certain IoT system with other devices or systems. This approach reduces the attack surface of the system and, consequently, the probability of cybersecurity risks. Furthermore, we complement this mechanism with a detection approach, which uses a distributed anomaly detection scheme based on deep learning (DL) and graph networks [17, 18]. Localized anomaly detection methods at IoT ports and network routers can also be investigated [19]. The mitigation method we proposed in this paper exploits network Self-Awareness [20, 21] centered on Software Defined Networks [22] that can achieve secure and QoS-based routing of significant traffic flows using machine learning and adaptivity [23–26].

The structure of the chapter is as follows: Section 5.2 analyzes the main processes composing a risk management and the challenges associated to the IoT paradigm. Then, the description of the Autopolicy approach is described in Section 5.3. Furthermore, Section 5.4 provides a detailed description of our approach for the detection of attacks and anomalies in IoT-based DL and graph networks.

5.2 Risk Management in IoT

The process of risk management is composed of different elements [13] including risk assessment and the definition of risk mitigation solutions. In particular, according to the definition provided by the National Institute of Standards and Technology (NIST), the risk management includes *"(i) the conduct of a risk assessment; (ii) the implementation of a risk mitigation strategy; and (iii) employment of techniques and procedures for the continuous monitoring of the security state of the information system"* [27]. Therefore, risk management implies the assessment, mitigation, and monitoring of risks.

Risk assessment is defined in CNSSI-4009 [28] as the process of *"identifying, prioritizing, and estimating risks"*. This includes determining the extent to which adverse circumstances or events could affect an enterprise. However, as described by [14, 29], risk assessment in IoT could be more difficult to implement in comparison to traditional ICT systems because of the pervasive connectivity of the IoT devices and the different potential interfaces (e.g., different wireless standards or middlewares), which can increase the attack surface. The assessment of the impact can also be critical in specific categories of IoT devices like the cyberphysical systems (e.g., an automated vehicle) or healthcare (e.g., insulin distributor) as a cybersecurity threat can have quite damaging consequences. Another challenge of the IoT domain for risk assessment is that IoT systems (e.g., a smart home or a smart

vehicle) can be composed of many different devices that, in turn, could be comprised of several components with a different security level. This means that the risk assessment of a certain system will depend on the security level associated to each part of the system. For this aspect, the main challenge is related to the way in which each risk value could be aggregated to provide a reliable value for the whole system.

A number of risk assessment methodologies have been developed in the literature and applied in some cases to the IoT domain. For example, the Common Weakness Scoring System (CWSS) [30] is a methodology to prioritize software weaknesses. The main motivation is to provide means to different stakeholders (e.g., software testers, or manufacturers) for quantifying the risks associated to a specific weakness. This way, the corresponding stakeholder can prioritize the weakness to be solved based on the estimated risk. Various metrics (i.e., Base Finding, Attack Surface and Environmental metric groups, which in turn contain different metric to quantify the CWSS score associated to a certain weakness) are used to produce a final CWSS score between 0 and 100. The main challenge is obviously the quantification process as cybersecurity risk is more difficult to quantify especially in the IoT domain [14]. A similar approach is also used in the Common Vulnerability Scoring System collection (CVSS) [31], which is based on three group metrics: Base, Temporal, and Environmental. The CVSS is widely used today as it is used in the National Vulnerability Database (NVD) created by the NIST. In the IoT literature, CVSS and CWSS schemes are used to assess the risks in Bluetooth technology, where the authors extended the authentication metric to include new security factors inherent to the Bluetooth technology [32]. CVSS and CWSS have also been used as an input in the process of cybersecurity certification of IoT devices in [33, 34]. Finally, another quantitative risk assessment scheme is DREAD, which is used to compute a risk value associated to a certain threat or vulnerability based on the use of five categories: Damage potential, Reproducibility, Exploitability, Affected users, and Discoverability. It was used at Microsoft, and currently, it is used by OpenStack. In [35], DREAD is applied to the mobile healthcare domain due to its simplicity.

As already mentioned, *risk identification* is a core process of a risk assessment approach. In this direction, threat and vulnerability assessments are usually employed to identify risks to organizational operations, and the evaluation of those risks in terms of likelihood of occurrence and impact if they occur. In turn, for the identification of threats and vulnerabilities, Intrusion Detection System (IDS) have been widely used [36] by monitoring and analyzing the network and/or system events. However, previous considerations make the application of well-known IDSs more challenging into IoT systems. Indeed, one of the popular approaches in literature is to analyze the data produced by IoT device to identify potential anomalies. A potential weakness of this approach is the difficulty to anticipate the

attack as the anomaly is created when the attack is already ongoing or it is completed. Moreover, machine learning capabilities have been increasingly become more powerful in recent times, and a complex security attack may be composed by a sequence of attacks steps. Then, the research objective is to the detect the initial steps in the attack.

These aspects are already highlighted by [37], which provides a comprehensive taxonomy of IDSs for IoT. Authors classify IDSs according to different parameters, such as the *placement strategy* (distributed, centralized, and hybrid) and *detection method* (signature-based, anomaly based, specification-based, and hybrid). In particular, anomaly based detection techniques are used to detect new attacks by comparing the normal behavior of a certain system with its actual activity. This approach could leverage the inherent nature of IoT systems, which are usually composed of special purpose devices. This aspect is considered by the recent Manufacturer Usage Description (MUD) standard [38], which is aimed to restrict the communication to/from a certain IoT device. To generate such intended behavior, anomaly detection techniques have widely used machine learning techniques, and in [39], a distributed anomaly detection approach in which each node monitors its neighbors is proposed, where monitoring nodes inform other nodes about security problems. In [40], deep autoencoders to detect anomalous network traffic of IoT devices are discussed and tested for different commercial IoT devices in the presence of IoT botnets such as Mirai. In our case, we describe a risk monitoring approach based on a multi-agent system and DL techniques for automatic feature extraction and anomaly detection that is described in Section 5.4.

As the next step of a risk management approach, *risk mitigation* focuses on the goal to mitigate the risks through different means: (a) avoiding the risk (e.g., not using a specific interface), (b) reducing the risk by implementing specific countermeasures (e.g., intrusion detection), or (c) transfer the risk to some other entity (e.g., an insurance company). In the IoT domain, the mitigation of risk is more difficult to achieve because of the limited computing capabilities of IoT devices, which makes more difficult the implementation of sophisticated countermeasures. The transfer of the risks is also difficult as IoT devices are often standalone systems (e.g., a sensor). A potential approach to mitigate IoT security risks is the integration of Software-Defined Network (SDNs), in order to restrict communications from/to IoT devices. Indeed, the application of SDN techniques into IoT scenarios has attracted a significant interest from academia in recent years. For example, [41] describes an architecture for managing the obtaining and enforcement of MUD restrictions, which are enforced by SDN switches. In this direction, our approach is based on an architecture to identify and mitigate security risks based on the communication profiles of IoT devices that is described in the next section.

5.3 Autopolicy System

The deployment of IoT scenarios requires new approaches to manage cybersecurity risks throughout the life cycle of devices composing such scenarios. In recent years, diverse proposals have been developed to address risk management aspects, in order to realize the best security practices for IoT (such as those defined by the (ENISA) [42]). In this direction, [43] proposed a system for automatically identifying the type of IoT devices that are connected to a network and enforcing security rules to restrict the communication of potentially vulnerable devices. Indeed, authors proposed the use of SDN techniques for the enforcement of such rules. Furthermore, [44] designed an automated approach to derive network security policies, as well as a multi-layered policy enforcement architecture.

Based on the considerations from previous works, our approach (Autopolicy) addresses several aspects of risk management in IoT scenarios. On the one hand, it reduces the attack surface and, consequently, the potential security risks associated to IoT devices by enforcing traffic profiles to be defined either by the device's manufacturer, or automatically. On the other hand, it links the obtaining of such profile to the authentication process of the IoT device, so that only traffic profiles from legitimate devices are obtained and enforced. Furthermore, it integrates a decentralized mechanism based on a Multi-agent System (MAS) for risk monitoring that is described in the next section. This way, Autopolicy helps to identify and monitor risks in a certain IoT network through traffic profiles, in order to mitigate the impact and likelihood associated to well-known threats, such as DoS attacks.

Autopolicy follows a similar approach to the recent IETF standard Manufacturer Usage Description (MUD) [38], which defines an architecture and data format to obtain and define network profiles for IoT devices. Indeed, MUD has received an increasing interest from other standardization organizations, such as the NIST, which recently published a Cybersecurity Practice Guideline [45] describing the advantages provided by MUD to reduce the potential harm of compromised devices. It follows a simple data model to generate network traffic rule allowing/denying the communication to/from a certain IoT device. In our case, we consider additional aspects, such as the maximum number of connections or restrictions on the message size, in order to properly react against DoS attacks.

An overview of the Autopolicy design is presented in Figure 5.1. The basic flow of information starts when a new *IoT device* joins a network, and it is detected by a switch (or controller) acting as *Device Authenticator*, which is responsible for granting/denying the access to such device. This initial authentication process is usually called *bootstrapping* [46], and it is used to exchange the corresponding cryptographic material between the IoT device and controller for authentication purposes.

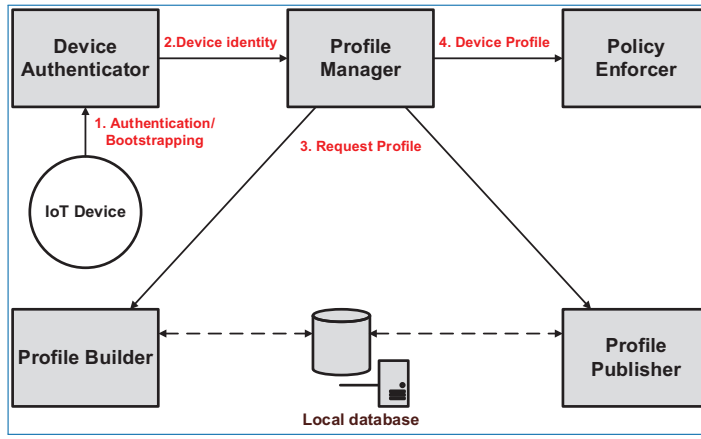


Figure 5.1. Functional diagram of Autopolicy and of its information flows.

For this purpose, there is a plethora of mechanisms that could be considered to instantiate this process, for example, based on the use of the Extensible Authentication Protocol (EAP) [47], which is widely considered in the scope of 5G networks. An alternative approach could be based on the OMA LwM2M specification [48], which explicitly defines a *Bootstrap Interface*, so that a LwM2M Client can be registered into the corresponding LwM2M Server. (OMA) defines four bootstrapping modes (factory, smartcard, client-initiated and server-initiated), and the use of transport layer security (i.e., TLS/DTLS) and application layer security based on the recent Object Security for Constrained RESTful Environments (OSCORE) [49].

After the bootstrapping process, the entity acting as Device Authenticator sends the authenticated device identity to the *Profile Manager*, which uses the identity to request the associated traffic profile. For example, following the MUD approach, the identity could be represented by a X.509 certificate in which a URL is included to get the traffic profile. In our case, the Profile Manager contacts the *Profile Publisher*, which manages the traffic profiles of IoT devices. Indeed, this entity follows a similar role to the MUD File Server [38], which is an entity provided by the manufacturer of the device. An alternative approach for sharing traffic profiles is represented by the use of blockchain as a trusted, distributed and transparent repository of traffic profiles. This way, manufacturers are enabled to implement smart contracts to manage the creation and update of traffic profiles. It should be noted that the use of blockchain is also considered in [50] to store cybersecurity information of IoT devices, including MUD profiles. In case there is not an associated profile to the device, the Profile Manager contacts the *Profile Builder* entity to create such profile based on IP traffic statistics and, optionally, additional information from other repositories where that device is already deployed.

Table 5.1 Traffic profile example.

```
{     "from_device": {
            "rate": 0.1,
            "allowed_dst": [
                  "91.200.1.0/24 tcp/80,443",
            ],
            "connections":"10"
      }
      "to_device": {
            "rate": 0.1,
            "blocked_dst": [
                  "0/0",
            ],
}}
```

The obtained traffic profile is then sent to the *Policy Enforcer*, which only allows IP traffic under strict rules defined by the profile. The role of the Policy Enforcer could be embedded in the controller (or switch) acting as Device Authenticator. Listing 5.1 shows a simple example of traffic profile to define restrictions on the communications to/from the device. In particular, *rate* specifies the maximum bandwidth (in Mbps); *allowed_dst* and *blocked_dst* indicate the IP address, protocol, and port of allowed/denied communications. Furthermore, *connections* represents the maximum number of connections allowed with a certain device.

While this represents a simple example of traffic profile, we plan to extend this initial data model and align it with the MUD standard. The described mechanism was designed for the needs of the SerIoT project as one of the components to prevent and mitigate potential risks in IoT systems. Its role is to secure the network from the malicious endpoints and ensure that IoT devices behave according to specific traffic rules. It should be noted that the described mechanism is intended to reduce the attack surface by enforcing the rules associated to the intended behavior of IoT devices. However, it still requires a dynamic approach to continuously monitor the risks that can be identified through traffic analysis. For this purpose, next section provides an overview of the risk monitoring approach that is being implemented in the scope of the SerIoT project.

5.4 Towards Distributed Attack Detection

The recent proliferation of IoT technologies has given rise to a dramatic increase in the number of edge devices. More devices not only introduce more security vulnerabilities but also greatly increase the volume of transferred data that must be analyzed in order to detect and mitigate network anomalies. When undetected,

such anomalies can have a great impact on the robustness and Quality of Service (QoS) of the attacked IoT infrastructure. The main challenge for anomaly detection methods in today's IoT network is the analysis of the big amounts of data that arise from the growing number of IoT devices. Such large volumes of information, as well as the distributed, large-scale, and heterogeneous nature of the IoT network, render the goal of (near) real-time anomaly detection difficult to accomplish using traditional centralized monitoring approaches [51].

In this respect, this work proposes the use of Multi-agent Systems (MAS) for processing the large amounts of data exchanged by the IoT devices in a distributed manner. The MAS allows for monitoring of the network traffic using parallel computation, resulting in faster detection times, and thus, improving the security and QoS of the IoT network. Additionally, when the agents have redundant roles (i.e., detection of anomalies regarding the same IoT device by different agents), the MAS system offers robustness, since it can tolerate failures of one or multiple agents [52]. Finally, MAS also offers scalability, since due to the modular nature of the approach, adding new agents to the network is easy and straightforward.

The main challenge when implementing MAS is how the agents will exchange information in order to solve a problem in a cooperative manner [53]. In this respect, this work utilizes recent advances in AI and Deep Learning (DL) in order to enable automatic feature extraction and anomaly detection by the agents of the MAS. Specifically, due to the graph structure of the network communication events, where nodes represent devices and edges communication, it is natural to consider DL approaches that deal with graph structured datasets, i.e., Graph Neural Network (GNN) [18]. This work utilizes as an inspiration the more generic GNN formulation, proposed by DeepMind [17].

The IoT network is defined as a graph $G(V, E, f_v, f_e)$, where each node $v_i \in V$ represents either an IoT device or a router, and each directed edge $e_j \in E$ represents a directed communication between nodes. Each node and edge is augmented with a feature vector through functions $f_v : V \rightarrow \mathbb{R}^{N_v}$ and $f_e : E \rightarrow \mathbb{R}^{N_e}$, respectively, where N_v and N_e, the size of node and edge feature vectors, respectively. These features are calculated using network traffic over a specific time window ΔT. The list of utilized features is presented in Table 5.1.

The agents of the MAS are installed in a subset of the nodes $V' \subseteq V$ and exchange information using the same set of communication edges E. Each agent comprised of two networks utilized for updating the feature representation of the adjacent edges and the node they are installed on. These updated features are afterwards utilized for multi-class anomaly detection through another pair of deep learning networks. An overview of this formulation for updating feature representation through information exchange between the different agents is illustrated in Figure 5.2. Under this consideration, the role of the edge Deep Neural Network

Table 5.1. The list of network features used for anomaly detection. The number of features for each node is $N_v = 5$ and for each edge is $N_e = 3$.

#	Target	Feature description
1	edge/node	Average number of packets sent
2	node	Average number of packets received
3	edge/node	Average number of bytes sent
4	node	Average number of bytes received
5	edge/node	Average connection duration

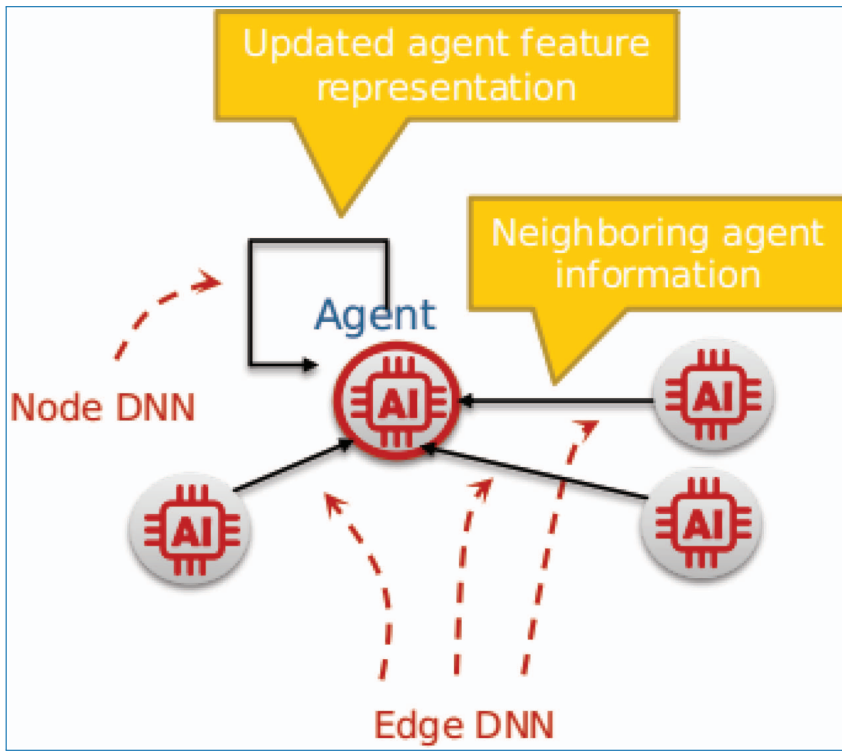


Figure 5.2. The procedure for updating the feature representations of the edges and nodes visible by each agent. The edge Deep Neural Network (DNN) takes as input the previous edge feature values and the adjacent agent features, and updates the feature representation of each edge. The node DNN aggregates the information from the updated edge feature values, and updates the feature representation of the specific node.

(DNN) is to integrate information from adjacent nodes, as well as the corresponding edge, and update this edge's feature representation. These updated feature representations for all adjacent edges are afterwards taken as input by the node DNN that updates the feature representation of the corresponding node. The Node and

Edge DNNs are the same for all agents (i.e., have shared weights), and thus, each agent is able to learn from events happening to other agents.

Based on the previous definitions, there are in total four DNNs, two used for updating node and edge feature representations, namely MLP_n and MLP_e, and two used for classification of neighbor nodes and the node in which the agent is installed, namely $Classifier_{neighbor}$ and $Classifier_{node}$.

Given an agent installed in node v_j, the procedure followed for updating the features of this node and the adjacent edges, as well as detecting anomalies in the entire neighborhood is formulated as follows:

$$\begin{aligned} e'^{i}_{\Delta T} &= MLP_e(x^{j}_{\Delta T}, x^{i}_{\Delta T}, e^{i}_{\Delta T}) \\ x'^{j}_{\Delta T} &= MLP_n\left(\frac{1}{N}\sum_{i=1}^{N} concat(e'^{i}_{\Delta T}, x^{i}_{\Delta T})\right) \\ p_i &= Classifier_{neighbor}(e'^{i}_{\Delta T}, x^{i}_{\Delta T}) \\ p_j &= Classifier_{node}(e'^{i}_{\Delta T}, x^{i}_{\Delta T}) \end{aligned} \tag{5.1}$$

where x^j is the feature vector of node v_j where the agent is installed, x^i is the feature of the neighboring nodes in the graph, e^i is the feature vector of the adjacent edges, and N is the number of neighboring nodes. This computation is performed with traffic data from a specific window ΔT. The MLP_n network is applied on the average of the updated features of each adjacent edge and node. The $concant(.)$ function takes as input two vectors and returns a larger vector which represents the concatenation of the two vectors. Equation (5.1) represents a single iteration of information exchange between agents. Stacking multiple blocks of such operations, the agents can exchange information multiple times within each time period ΔT. The architecture of this procedure is illustrated in Figure 5.3. The entire network is trained in a supervised manner using the back-propagation algorithm with cross-entropy as loss for multi-class classification.

In this formulation of MAS, each agent is responsible for performing anomaly detection not only on itself but also on its neighboring nodes. This redundancy of the role of the agents enables the system to be robust in cases where one or multiple agents may fail, since other agents will take over the role of detecting anomalies in neighboring nodes instead of them. The issue that arises, however, is how to combine the overlapping decisions of the different agents into a single decision for each node in the IoT network. This work utilizes a simple aggregation method, where a node is considered anomalous if at least one agent reported it as

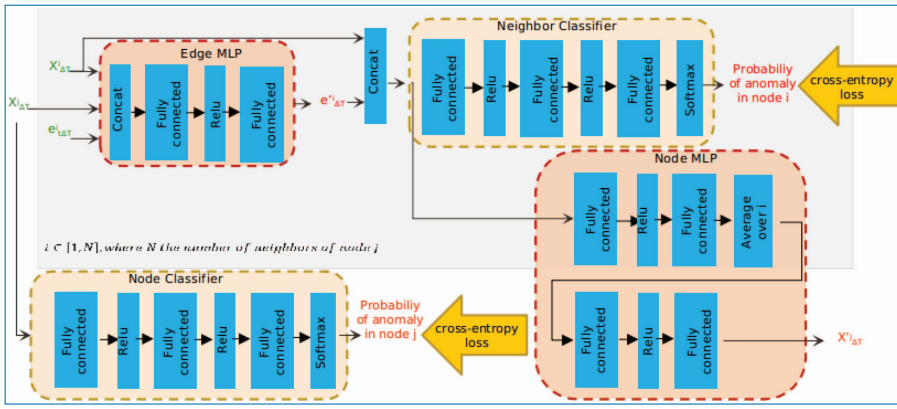


Figure 5.3. The architecture of each DNN agent in the multi-agent anomaly detection system. The edge Multi-layer Perception (MLP) takes information from the N neighboring agents and updated the values of the edge features, while the node MLP combines the updated edge feature values and updates the feature values of the specific node (agent). Edge features are used for predicting anomalies in the neighborhood, while node features are used for detecting anomalies in the specific node, using the classifier networks. The training is performed in a supervised manner using back-propagation with cross-entropy loss.

anomalous. Alternative more sophisticated aggregation schemes will be considered in future work.

5.5 Conclusions

The implementation of a suitable risk management approach demands for holistic approaches for the assessment, mitigation, and monitoring of security risks in IoT systems. Toward this end, we have provided an initial description of a policy-based system that is being defined in the scope of the EU SerIoT project. Our approach is based on the use of network traffic profiles, which specify the intended communications of IoT devices. Such effort is aligned with the recent MUD standard, which has received an increasing interest from academia and industry during last year. Autopolicy defines an architecture for obtaining and enforcing traffic profiles, in order to mitigate potential security risks in IoT systems. Furthermore, it integrates a distributed machine learning approach for risk monitoring by analyzing the network traffic. As a future work, we plan to adopt the MUD standard as the baseline to define and extend our traffic profiles to be enforced through the components defined in the SerIoT SDN architecture. Furthermore, we will provide an implementation and detailed description of our adaptive machine learning approach to reroute IoT traffic according to the identification of compromised nodes of a certain network.

Acknowledgments

This research is supported by the European Commission H2020-IOT-2016-2017 (H2020-IOT-2017) Program under Grant Agreement 780139 for the SerIoT Research and Innovation Action.

References

[1] Bera, S., Misra, S., and Vasilakos, A.V.: Software-defined networking for internet of things: A survey. IEEE Internet of Things Journal **4**(6), 1994–2008 (2017)

[2] Elhammouti, H., Sabir, E., Benjillali, M., Echabbi, L., and Tembine, H.: Self-organized connected objects: Rethinking QoS provisioning for IoT services. IEEE Communications Magazine **55**(9), 41–47 (2017)

[3] Melcherts, H.E.: The internet of everything and beyond. Human Bond Communication: The Holy Grail of Holistic Communication and Immersive Experience p. 173 (2017)

[4] Ramos, J.L.H., Geneiatakis, D., Kounelis, I., Steri, G., and Fovino, I.N.: Toward a data-driven society: A technological perspective on the development of cybersecurity and data protection policies. IEEE Security & Privacy (2019)

[5] Gelenbe, E., Gorbil, G., Tzovaras, D., Liebergeld, S., Garcia, D., Baltatu, M., and Lyberopoulos, G.: Security for smart mobile networks: The nemesys approach. In: Privacy and Security in Mobile Systems (PRISMS), 2013 International Conference on. pp. 1–8. IEEE (2013)

[6] Ratasuk, R., Prasad, A., Li, Z., Ghosh, A., and Uusitalo, M.A.: Recent advancements in M2M communications in 4G networks and evolution towards 5G. In: Proc. 18th IEEE International Conference Intelligence in Next Generation Networks (ICIN). pp. 52–57. Paris, France (Feb 2015). https://doi.org/10.1109/ICIN.2015.7073806

[7] Collen, A. *et al.*: Ghost: Safeguarding home IoT environments with personalised real-time risk control. In: Recent Cybersecurity Research in Europe: Proceedings of the 2018 ISCIS Security Workshop, Imperial College London. vol. LNCCIS 821. Springer Verlag (2018)

[8] Gelenbe, E. and Kadioglu, Y. M.: Energy life-time of wireless nodes with network attacks and mitigation. 2018 IEEE International Conference on Communications Workshops (ICC Workshops), Kansas City, MO, pp. 1–6, (2018). doi: 10.1109/ICCW.2018.8403561.

[9] He, D., Chan, S., Qiao, Y., and Guizani, N.: Imminent communication security for smart communities. IEEE Communications Magazine **56**(1), 99–103 (Jan 2018). https://doi.org/10.1109/MCOM.2018.1700587

[10] Kalkan, K. and Zeadally, S.: Securing internet of things (IoT) with software defined networking (sdn). IEEE Communications Magazine (2017). https://doi.org/10.1109/MCOM.2017.1700714
[11] Lu, X., Spear, M., Levitt, K., Matloff, N.S., and Wu, S.F.: A synchronization attack and defense in energy-efficient listen-sleep slotted MAC protocols. In: Emerging Security Information, Systems and Technologies, 2008. SECURWARE'08. Second International Conference on. pp. 403–411. IEEE (2008)
[12] Pirretti, M., Zhu, S., Vijaykrishnan, N., McDaniel, P., Kandemir, M., and Brooks, R.: The sleep deprivation attack in sensor networks: Analysis and methods of defense. International Journal of Distributed Sensor Networks **2**(3), 267–287 (2006)
[13] Purdy, G.: Iso 31000: 2009 — setting a new standard for risk management. Risk Analysis: An International Journal **30**(6), 881–886 (2010)
[14] Nurse, J.R., Creese, S., and De Roure, D.: Security risk assessment in internet of things systems. IT Professional **19**(5), 20–26 (2017)
[15] Sen, S., Koo, J., and Bagchi, S.: Trifecta: Security, energy-efficiency, and communication capacity comparison for wireless IoT devices. IEEE Internet Computing **22**(1), 74–81 (2018)
[16] Gelenbe, E., Domanska, J., Czachórski, T., Drosou, A., and Tzovaras, D.: Security for internet of things: The SerIoT project. In: 2018 International Symposium on Networks, Computers and Communications, ISNCC 2018, Rome, Italy, June 19-21, 2018. pp. 1–5. IEEEXplore (2018), https://doi.org/10.1109/ISNCC.2018.8531004
[17] Battaglia, P., Hamrick, J.B.C., Bapst, V., Sanchez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., Gulcehre, C., Song, F., Ballard, A., Gilmer, J., Dahl, G.E., Vaswani, A., Allen, K., Nash, C., Langston, V.J., Dyer, C., Heess, N., Wierstra, D., Kohli, P., Botvinick, M., Vinyals, O., Li, Y., and Pascanu, R.: Relational inductive biases, deep learning, and graph networks. arXiv (2018), https://arxiv.org/pdf/1806.01261.pdf
[18] Zhang, Z., Cui, P., and Zhu, W.: Deep learning on graphs: A survey. arXiv preprint arXiv:1812.04202 (2018)
[19] Brun, O., Yin, Y., and Gelenbe, E.: Deep learning with dense random neural network for detecting attacks against IoT-connected home environments. Procedia Computer Science **134**, 458–463 (2018)
[20] Gelenbe, E., Liu, P., and Lainé, J.: Genetic algorithms for route discovery. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) **36**(6), 1247–1254 (2006)
[21] Mehmood, Y., Ahmad, F., Yaqoob, I., Adnane, A., Imran, M., and Guizani, S.: Internet-of-things-based smart cities: Recent advances and challenges. IEEE Communications Magazine **55**(9), 16–24 (2017)

[22] Francois, F. and Gelenbe, E.: Towards a cognitive routing engine for software defined networks. In: Communications (ICC), 2016 IEEE International Conference on. pp. 1–6. IEEE (2016)
[23] Dobson, S. *et al.*: A survey of autonomic communications. ACM Transactions on Autonomous and Adaptive Systems (TAAS) **1**(2), 223–259 (2006)
[24] Galis, A., Denazis, S., Brou, C., and Klein, C.: Programmable Networks for IP Service Deployment. Artech House Inc. (2004)
[25] Rubio-Loyola, J., Astorga, A., Serrat, J., Chai, W.K., Mamatas, L., Galis, A., Clayman, S., Cheniour, A., Lefevre, L., Mornard, O., Fischer, A., Paler, A., and Meer, H.D.: Platforms and software systems for an autonomic internet. In: 2010 IEEE Global Telecommunications Conference GLOBECOM. IEEE (2010)
[26] Tsarouchis, C., Denazis, S., Kitahara, C., Vivero, J., Salamanca, E., Magana, E., Galis, A., Manas, J.L., Carlinet, L., Mathieu, B., and Koufopavlou, O.: A policy-based management architecture for active and programmable networks. IEEE Network **17**(3), 22–28 (2003)
[27] Joint Task Force Transformation Initiative: Guide for applying the risk management framework to federal information systems: a security life cycle approach. Tech. Rep. NIST SP 800-37r1, National Institute of Standards and Technology (Jun 2014). https://doi.org/10.6028/NIST.SP.800-37r1, https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-37r1.pdf
[28] USC, S.: 3502. CNSSI-4009
[29] Radanliev, P., De Roure, D.C., Nicolescu, R., Huth, M., Montalvo, R.M., Cannady, S., and Burnap, P.: Future developments in cyber risk assessment for the internet of things. Computers in Industry **102**, 14–22 (2018)
[30] Martin, B. and Coley, S.: Common weakness scoring system (CWSS). Internet, http://cwe.mitre.org/cwss (2014)
[31] Mell, P., Scarfone, K., and Romanosky, S.: Common vulnerability scoring system. IEEE Security & Privacy **4**(6), 85–89 (2006)
[32] Qu, Y. and Chan, P.: Assessing Vulnerabilities in Bluetooth Low Energy (BLE) Wireless Network Based IoT Systems. In: 2016 IEEE 2nd International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS). pp. 42–48. IEEE, New York, NY, USA (Apr 2016). https://doi.org/10.1109/BigDataSecurity-HPSC-IDS.2016.63, http://ieeexplore.ieee.org/document/7502262/

[33] Matheu-Garcia, S.N., Hernandez-Ramos, J.L., and Skarmeta, A.F.: Test-based risk assessment and security certification proposal for the Internet of Things. In: 2018 IEEE 4th World Forum on Internet of Things (WF-IoT). pp. 641–646. IEEE, Singapore (Feb 2018). https://doi.org/10.1109/WF-IoT.2018.8355193, https://ieeexplore.ieee.org/document/8355193/

[34] Matheu-Garcia, S.N., Hernandez-Ramos, J.L., Skarmeta, A.F., and Baldini, G.: Risk-based automated assessment and testing for the cybersecurity certification and labelling of IoT devices. Computer Standards & Interfaces **62**, 64–83 (Feb 2019). https://doi.org/10.1016/j.csi.2018.08.003, https://www.sciencedirect.com/science/article/abs/pii/S0920548918301375?via%3Dihub

[35] Cagnazzo, M., Hertlein, M., Holz, T., and Pohlmann, N.: Threat modeling for mobile health systems. In: 2018 IEEE Wireless Communications and Networking Conference Workshops (WCNCW). pp. 314–319. IEEE, Barcelona (Apr 2018). https://doi.org/10.1109/WCNCW.2018.8369033, https://ieeexplore.ieee.org/document/8369033/

[36] Debar, H., Dacier, M., and Wespi, A.: Towards a taxonomy of intrusion-detection systems. Computer Networks **31**(8), 805–822 (1999)

[37] Zarpelao, B.B., Miani, R.S., Kawakani, C.T., and de Alvarenga, S.C.: A survey of intrusion detection in internet of things. Journal of Network and Computer Applications **84**, 25–37 (2017)

[38] Lear, E., Romascanu, D., and Droms, R.: Manufacturer Usage Description Specification (RFC 8520) (2019), https://tools.ietf.org/html/rfc8520

[39] Thanigaivelan, N.K., Nigussie, E., Kanth, R.K., Virtanen, S., and Isoaho, J.: Distributed internal anomaly detection system for internet-of-things. In: 2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC). pp. 319–320. IEEE (2016)

[40] Meidan, Y., Bohadana, M., Mathov, Y., Mirsky, Y., Shabtai, A., Breitenbacher, D., and Elovici, Y.: N-BaIoT network-based detection of IoT botnet attacks using deep autoencoders. IEEE Pervasive Computing **17**(3), 12–22 (2018)

[41] García, S.N.M., Molina Zarca, A., Hernández-Ramos, J.L., Bernabé, J.B., and Gómez, A.S.: Enforcing behavioral profiles through software-defined networks in the industrial internet of things. Applied Sciences **9**(21), 4576 (2019)

[42] ENISA: Good Practices for Security of Internet of Things in the context of Smart Manufacturing (2018), https://www.enisa.europa.eu/publications/good-practices-for-security-of-IoT

[43] Miettinen, M., Marchal, S., Hafeez, I., Asokan, N., Sadeghi, A.R., and Tarkoma, S.: IoT sentinel: Automated device-type identification for security enforcement in IoT. In: 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS), 5–8 June 2017. pp. 2177–2184. IEEE (2017). https://doi.org/10.1109/ICDCS.2017.284

[44] Barrera, D., Molloy, I., and Huang, H.: Standardizing IoT network security policy enforcement. In: Workshop on Decentralized IoT Security and Standards 2018 (01 2018). https://doi.org/10.14722/diss.2018.23007, http://wp.internetsociety.org/ndss/wp-content/uploads/sites/25/2018/07/diss2018_7_Barrera_paper.pdf

[45] NIST: Securing Small-Business and Home Internet of Things Devices: NIST SP 1800-15 (2019)

[46] Garcia-Morchon, O., Kumar, S.S., and Sethi, M.: Internet of Things Security: State of the Art and Challenges (RFC 8576) (2019)

[47] Simon, D., Ph.D., D.B.D.A., and Eronen, P.: Extensible Authentication Protocol (EAP) Key Management Framework. RFC 5247 (Aug 2008). https://doi.org/10.17487/RFC5247, https://rfc-editor.org/rfc/rfc5247.txt

[48] Rao, S., Chendanda, D., Deshpande, C., and Lakkundi, V.: Implementing lwm2m in constrained IoT devices. In: 2015 IEEE Conference on Wireless Sensors (ICWiSe). pp. 52–57. IEEE (2015)

[49] Selander, G., Mattsson, J., Palombini, F., and Seitz, L.: Object security for constrained restful environments (oscore) (July 2019), https://tools.ietf.org/html/rfc8613

[50] Neisse, R., Hernández-Ramos, J.L., Matheu, S.N., Baldini, G., and Skarmeta, A.: Toward a blockchain-based platform to manage cybersecurity certification of IoT devices. In: 2019 IEEE Conference on Standards for Communications and Networking (CSCN). pp. 1–6. IEEE (2019)

[51] Khodadadi, F., Dastjerdi, A.V., and Buyya, R.: Internet of things: an overview. In: Internet of Things, pp. 3–27. Elsevier (2016)

[52] Qin, J., Ma, Q., Shi, and Y., Wang, L.: Recent advances in consensus of multi-agent systems: A brief survey. IEEE Transactions on Industrial Electronics **64**(6), 4972–4983 (2016)

[53] Gupta, J.K., Egorov, M., and Kochenderfer, M.: Cooperative multi-agent control using deep reinforcement learning. In: International Conference on Autonomous Agents and Multiagent Systems. pp. 66–83. Springer (2017)

DOI: 10.1561/9781680836837.ch6

Chapter 6

Chariot-integrated Approach to Safety, Privacy, and Security – CHARIOT IPSE

By Aydin Ulas, Bora Caglayan, Sofiane Zemouri, George Theofilis, Konstantinos Loupos, Antonis Mygiakis, Andrea Battaglia, Mario Villiani, Christos Skoufis and Stelios Christofi

Industrial IoT applications in various domains including large office sites and transport infrastructure are getting more complex with multiple computer-based management systems handling the workload. These systems can generate data that can be used to introduce cognitive capability in many critical tasks. Handling the concerns related to privacy and security is important to enable any complex functionality [1–3]. For this reason, the CHARIOT IPSE (IoT Privacy and Safety Engine) shows a solution that links the privacy, security, and safety supervision functionality in a novel way.

In summary, the safety supervision engine will propose a mechanism to process and handle the signals generated by sensors for safety-related scenarios. Lately, IoT deployments are moving away from an approach of offloading the cognitive capabilities to one central approach towards distributed intelligence with multiple nodes handling the cognitive capabilities of the IoT deployment [1, 4]. The implementation of machine learning-based policies and propagating the changes on the systems require edge cloud orchestration done by the system [5, 6]. To align with the CHARIOT objectives, CHARIOT develops a safety supervision

engine that can run on multiple Fog nodes of an IoT setup. In addition, the safety supervision engine provides collaborative optimization capability to dynamically change priorities of sensor based on the operation mode of the system. By doing dynamic prioritization of signal transfer to cloud, the scalability issue which is seen as one of the major research challenges for industrial IoT [7] is tackled.

The CHARIOT Privacy engine uses blockchain to avoid privacy breaches in the IoT network. The Privacy engine encrypts the data on a southbound dispatcher (sensor side in the architecture) by using keys distributed on the blockchain network. It keeps the policies and the access rights for particularly sensitive data in the policy repository and enforces the access rights for CHARIOT. The Privacy engine is particularly suitable for large industrial IoT deployment environments where multiple contractors may access different data streams [8, 9].

The CHARIOT Security Engine checks the firmware updates and related changes in network and network transmission to make sure that the data shared in the system is bona fide with no security breaches [10, 11]. CHARIOT uses open source tools and heuristics where historical firmware binary data will be used to identify updates with security vulnerabilities and inform the operator.

In this chapter, we will provide an overview of the core capabilities of CHARIOT safety supervision engine, privacy engine, and security engine. An overview of the Engine's capabilities is also provided.

6.1 The CHARIOT Safety Supervision Engine

Industrial IoT systems are deployed in a wide range of applications in different domains. Three living labs (piloting cases in Airports, Rail, and Smart Buildings) included in CHARIOT cover a broad range of industrial IoT deployments in a broad range of domains including transport, logistics, and workplace management. The CHARIOT Safety Supervision Engine used the key pain points and business requirements of these industrial IoT deployments while specifying, designing, and implementing it.

State-of-the-art IoT safety supervision solutions including Microsoft Azure and IBM Watson IoT Platform handle widely any safety critical functionalities that might require complicated analysis on the Cloud Environment. Based on the extracted and focal Living Labs' industrial requirements, there is a clear need for deploying more functionality on the edge of the network, and a Cloud-only solution would not be sufficient. For example, an IoT deployment on a train may lose internet connectivity in a tunnel or an airport environment may lose internet

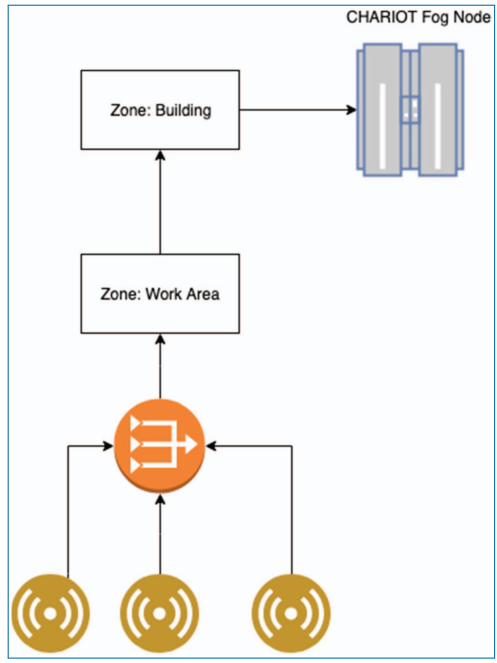


Figure 6.1. A sample industrial IoT topology for a work campus environment.

connection during a hazard. A solution should provide a form of fault tolerance to avoid single point of failure (Figure 6.1).

The CHARIOT safety supervision engine is designed to be flexible enough for any industrial IoT system. CHARIOT has defined three abstract entities in three layers. The layers can also have hierarchical structure among themselves:

- **Zone:** Zone is any physical area that includes some networked components. A zone might be a train (Trenitalia), building floor (IBM), or airport room (Athens Airport). A zone may be hierarchically attached to a parent zone (e.g., a room in a building floor). A zone may have an associated CHARIOT Fog Node. All the Fog nodes run the safety supervision engine separately to enable fault tolerance.
- **Gateway:** Gateway is any computational node that has network connection with one or more sensors. A gateway in CHARIOT can also be a fog node with relatively more computational power. These fog nodes can store the partial state of the IoT Setup.
- **Sensor:** A sensor is any networked component that can collect data. The sensor can be simple or complicated. A sensor can be optionally part of a gateway. This relation is optional since some sensors are complicated enough to have direct connection.

The three layers of elements can have hierarchical dependencies. For example, a zone might have two gateways connected to multiple sensors. CHARIOT uses an IoTL (Internet of Things Language – A domain-specific language developed for CHARIOT Project) to keep all these dependencies consistent across various nodes in the industrial IoT setup. The dependencies are kept consistent on any node that is keeping the state of the industrial IoT Setup. In addition, using IoTL, one can easily define custom types and hierarchies for custom scenarios.

IoTL is a language for communicating the state of the system. Its grammar is created using ANTLR, and a Python 3 parser is generated by ANTLR. IoTL is used to share and track the topology of an IoT setup across different components. This language enables communicating safety-related actions and state changes between multiple nodes of a system. The core specs of the IoTL are listed as follows:

1. **Definitions:** We use zone, gateway, and sensor as the key elements of an IoT network. A zone is an enclosed area in an industrial IoT setup which might have multiple zones, gateways, and sensors. Gateway is another core unit which might be running CHARIOT safety engine if it is Linux enabled. It controls the traffic of multiple sensors. A zone can be associated with a Fog node.
2. **Relations:** A relation is defined between two components in the system. The types of allowed relations are listed as follows:
   (a) Dependency: A dependency is between a master and slave component and denotes that the slave component is part of or controlled by the master component. When the system state is synced to a local component, the state associated with all the dependent components is shared too. For example, several gateways might exist in a zone or several zones might have a master zone.
   (b) Correlation: Correlation can exist only between sensor components. Violation of a correlation creates an abnormal state on the system. For example, a certain level of correlation can be enforced for temperature sensors in a room. If this policy is violated, the system may trigger fault sensor alert on the CHARIOT Platform.
   (c) Equality: Equality is equal recordings by two sensors at the same time stamp.
   (d) Delayed condition: Delayed condition is a condition applied within a time interval. For example, the presence of sensor in a corridor might be equal with a certain delay.

3. **Enforcements:** Enforcements are plugged machine learning models and policies defined for the Industrial IoT environment. A violation of an enforcement is classified as an alert by the alert manager.
4. **Alerts:** Alerts are next actions after an anomalous condition in the system.

The CHARIOT Safety Supervision Engine can run on multiple nodes within a distributed IoT deployment as seen in Figure 6.2. The nodes are associated with zones in the safety supervision model. In a deployment, more than one node can be deployed. Deployment of the same runtime on multiple instances ensures the system availability even when the network is partitioned during a safety hazard. The zone hierarchy is the same as the zone hierarchy in this setup. The safety supervision engine is a web application that provides a REST-APIs and connectors for live data streams. All the Safety Supervision Engine functionality is written on Python Ecosystem and using open source Python libraries. Flask framework was chosen due to its compactness, popularity, and easy integrability with machine learning functionality written in Python Language.

The CHARIOT Safety Supervision Engine can run on multiple networked nodes in complicated IoT deployments. In this case, multiple fog nodes can be associated with different zones. Based on the hierarchy and the dependencies, each fog node can observe only associated parts of the system, and they synchronize periodically to keep the state of the system consistent across the nodes. Every change

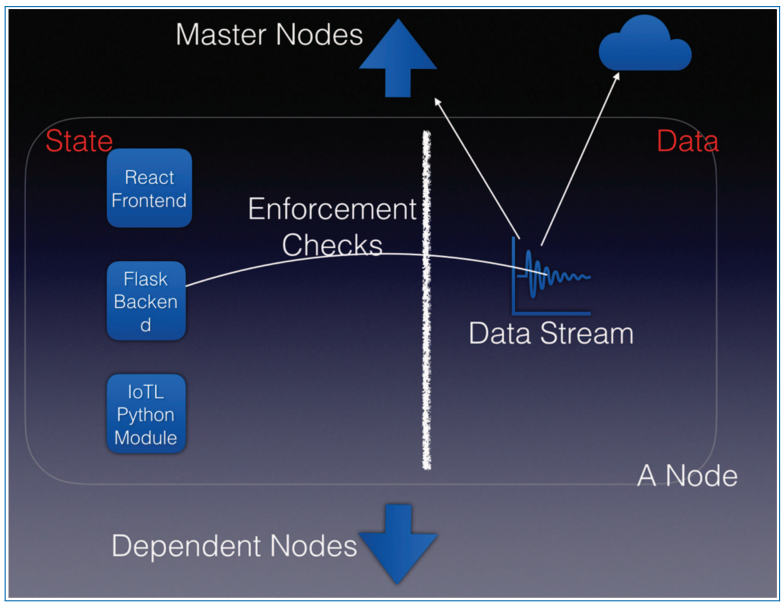


Figure 6.2. Safety supervision node example.

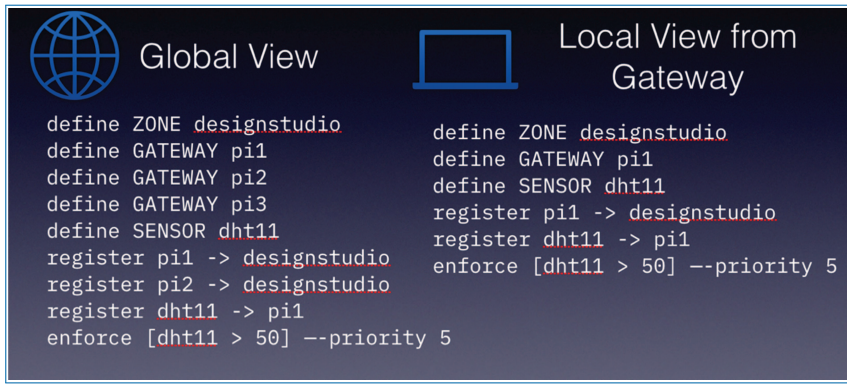


Figure 6.3. A simple example of global and local state in the safety supervision engine.

is hierarchically synchronized across all the computation nodes across the Fog Network using the commands API with the "!sync" command. The high-level representation of the nodes can be seen in Figure 6.2. The communication across the safety supervision platform instances is carried out. In the current living lab environments, only one Fog node exists for purposes of the CHARIOT proof of concept and industrial cases.

The dependency relation across zones is kept within the state of the IoT Deployment. In the case of synchronization across different fog nodes, only state relevant to one or more dependent component is shared with a fog node. For example, if there is no relevant change in the state for one fog node, the state is not updated to conserve resources. In Figure 6.3, we provide a simple example for this state sharing mechanism. In this simple scenario, the global view has a bunch of gateways associated with a single zone in an IoT deployment. The state relevant to the gateway named pi1 is kept by the fog node associated with that gateway. In the case of a state change, only the relevant portions to the element are shared. By definition, the global state associated with the root zone "designstudio" is associated with all the elements in the IoT deployment, so it receives all the updates. On the contrary, the fog node associated with pi1 only receives if a portion of the deployment relevant to it has been updated.

6.2 The CHARIOT Privacy Engine

"Privacy" has been a fundamental issue since the early days of computing, especially IoT. IoT is becoming part of everyday living and everything is being shared over internet—photos, videos, health measurements, gaming records, etc—privacy IS the area of matter of worry. In IoT, privacy has different scenario as compared to others as mode of data collection, mining, and provisioning is different. Technically,

data can be collected in such a manner that it may make difficult to interpret and realize that personal data are ready to be compromised. Privacy of individuals in IoT is a serious compromise factor. Within CHARIOT, privacy is given special attention by ensuring that the system has control over:

- Data being collected by individual sensors should enter the system if only the sensor is known and registered in the topology.
- If data are from a known sensor, data encryption must be applied using a public key stored in a blockchain PKI.

So, "Promoting Privacy and Data Protection principles remains paramount to ensure acceptance of IoT Services."

6.2.1 IoTL Language Extension—Access Control

The CHARIOT's IoTL was extended to include Access Control constructs that provide declarative access control over the elements of the CHARIOT Network.

The **Access Control IoTL extension** introduces the following definitions:

1. **Principal:** Defines the person or entity that has listened on CHARIOT for new messages. A principal example is an Industrial System Gateway.
2. **Resource:** Defines the things that the ACL rule applies to. Some examples are the Sensors, Logs, or the alerts.
3. **Operation:** Identifies the action that the rule governs. CHARIOT supports only **READ**.
4. **Guard:** Is the responsible service to apply ACL rules, for us the Privacy Engine.
5. **Action:** Identifies the action of the rule. It must be one of: **ALLOW, DENY**.

An indicative application example of access control for CHARIOT follows below:

The example (Figure 6.4) of an IoT network includes five sensors, a gateway and two external services. All the sensors are connected to the gateway and the external services are expecting data from sensors through the gateway. To define a public key, we must add two properties in a json parameter of the entity definition; these properties are **pubkey** and **type. Pubkey** is the public key data at any valid format and **type** is which encryption scheme is used by the service (Figure 6.5).

After all services, gateway, and sensor are defined, we can add the rules to allow communication between sensor and service. As seen in Figure 6.6, we allow **BMS** to get observations from **S2** but we forbid it to get observations from **S1**.

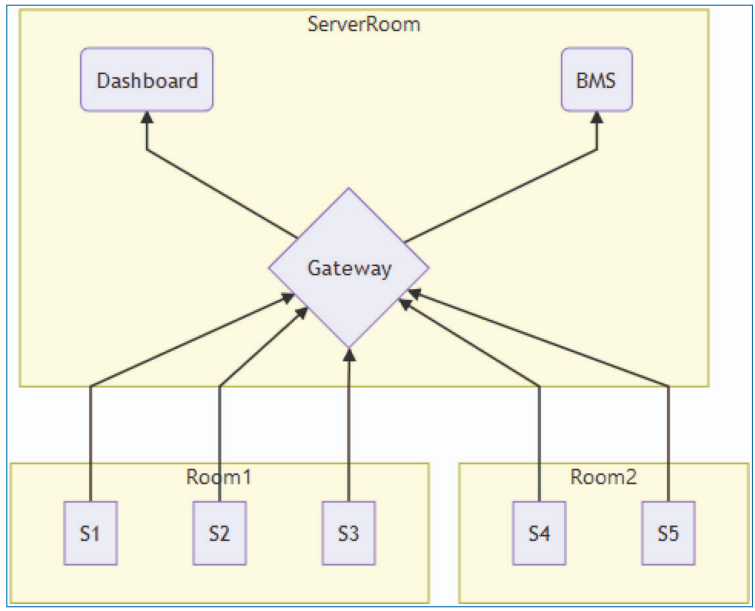


Figure 6.4. Sample network design.

```
define GATEWAY BMS --params { "pubkey": "-----BEGIN PUBLIC
KEY-----\nMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCwusoeNOkZh8gvX7BGEy+rhRxV\nF/ZD11xm0UpzfTR5k
/VTasjSyY1yzs2P0BePMUM78cJF21hEBL5fAFCqKpH7zhAj\nl5fFcQd/kZuIlB5ijJAjJhCKV8SK2rwXQXemo9Gc2PHdSg63qjYhEB55dPcClfNw
\nCoWsKkKI55WtVjKsDQIDAQAB\n-----END PUBLIC KEY-----", "type": "RSA" }
```

Figure 6.5. Define and external service and its public key.

```
acl BMS S1 DENY
acl BMS S2 ALLOW
```

Figure 6.6. Rule to REVOKE or GRANT privileges to BMS.

In this network, Room 2 must be protected by every unauthorized access, and each data leaving this room must be tracked. In this case, the Privacy Engine is expected to raise an alert for the administrator. We define this by using again the parameter statement of the IoTL. Sensor sensitivity declarations can be seen in Figure 6.7.

The components of the Privacy Engine (Figure 6.8) have been included below:

- **Listener**
  Listener is a component waiting for any sensor measurement—data in analogue or digital format, within the IoT system.

```
define SENSOR S1 --params { "privacySensitive": 0 }
define SENSOR S2 --params { "privacySensitive": 0 }
define SENSOR S3 --params { "privacySensitive": 0 }
define SENSOR S4 --params { "privacySensitive": 1 }
define SENSOR S5 --params { "privacySensitive": 0 }
```

Figure 6.7. Example of how we can define sensitive sensor.

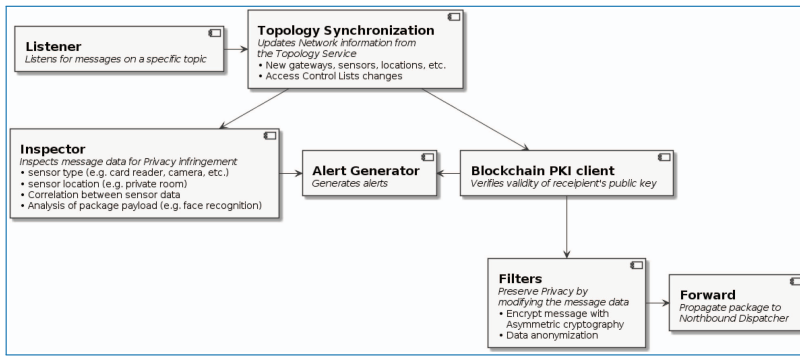

Listener
Listens for messages on a specific topic
Topology Synchronization
Updates Network information from the Topology Service
• New gateways, sensors, locations, etc.
• Access Control Lists changes
Inspector
Inspects message data for Privacy infringement
• sensor type (e.g. card reader, camera, etc.)
• sensor location (e.g. private room)
• Correlation between sensor data
• Analysis of package payload (e.g. face recognition)
Alert Generator
Generates alerts
Blockchain PKI client
Verifies validity of receipient's public key
Filters
Preserve Privacy by modifying the message data
• Encrypt message with Asymmetric cryptography
• Data anonymization
Forward
Propagate package to Northbound Dispatcher

Figure 6.8. Privacy engine components.

- **Topology Synchronization**
  This component is responsible to keep information describing new sensors topology within the IoT system and any update in sensor description (location, sensitivity, etc.). Also, all valid recipients should be registered in this component.
- **Inspector**
  Inspector is the component responsible to raise an Alert Flag every time a message is generated by a "Sensitive" labeled sensor, in order all actors within the IoT Network to be aware of that type of sensors. To achieve this, the network administrator should label a sensor as "Sensitive" so as to prevent a privacy information violation.
- **Blockchain PKI client**
  This component assures verification of the Public Key used by the recipient. If there is a breach, an alert will be raised.
- **Alert Generator**
  Alert Generator, if triggered, will generate an alert and will also keep record of all alerts.
- **Filter**
  CHARIOT Admin should define the Access List Controls with IoTL in order to guard user data. Filtering rule (Figures 6.9) will effectively avoid leakage of

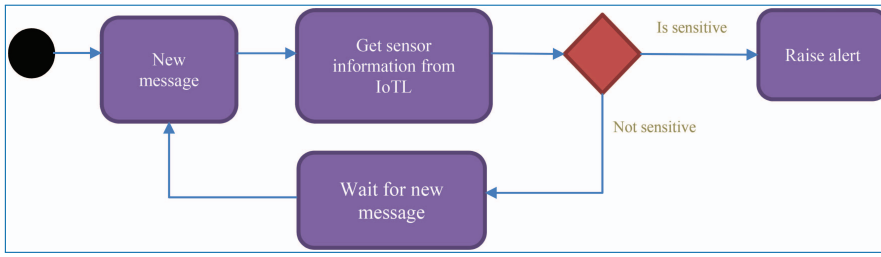


Figure 6.9. How topology-related filtering rules are applied.

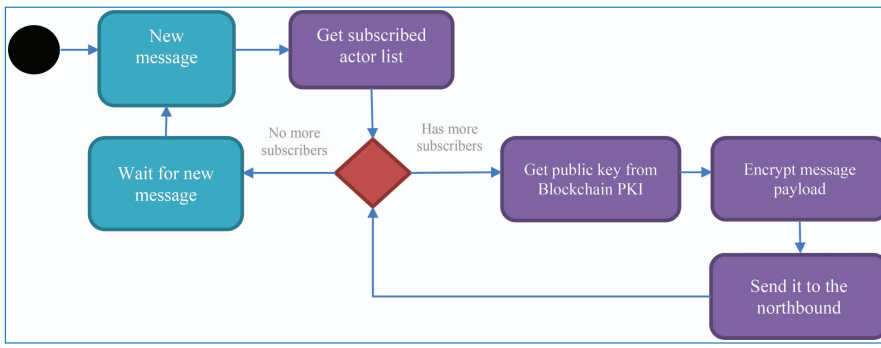


Figure 6.10. How cipher filtering rules are applied.

sensitive data to unauthorized users. Public-Key Encryption and Access Control List are used to achieve this. On the following flowchart (Figure 6.10), the process of public-key encryption of message payload is shown.

- **Northbound Dispatcher client**
  This component will propagate the final data package to the IoT network.

6.3 The CHARIOT Security Engine

In today's context, smart devices belonging to the Internet of Things ecosystem (IoT ecosystem) are increasingly present. However, the technology companies producing these evolved systems tend to underestimate the potential risks to which they are exposed to, since the latter are connected almost continuously to the internet. The greatest danger is caused by superficial behavior conducted by the manufacturers themselves who are oriented to release the firmware updates with an excessively low frequency and sometimes, fortunately only in extreme cases so far, the updates are not released at all. This superficial behavior allows hackers to take advantage of a vast amount of potentially vulnerable devices.

The firmware, an expression comprised of the terms firm (stable) and ware (component), is a program present within each electronic component that performs a critical task. Its fundamental role is to start the function of the same component allowing it to communicate with further integrals or cards present in the device in which they are installed. This determines the importance for releasing firmware updates; otherwise, an IoT device that has not been updated since its purchase can become dangerous for the consumer when cybercriminals discover a vulnerability of the system. This usually occurs when a device manufacturer decides to allocate its internal resources to the production of a new product to be introduced to the market, instead of the development and release of updated firmware versions for existing products. Consequently, the most significant risk for the consumer is finding obsolete and potentially dangerous security hardware.

Failure to update the software should not always be attributed to the manufacturer; the end user plays an active and harmful role. Quite often, due to personal inexperience or lack of knowledge for the "smart" product, the consumer does not check for updates and does not install the firmware versions released and recommended by the manufacturer. To overcome this particular situation and reduce the security risks related to the exposing the security of the software platforms, the most advanced devices, including computers, are programmed to perform automatic updates without the need of any customer intervention. Unfortunately, updating and protecting the security features of the systems belonging to the Internet of Things is not, and will not be, trivial to the implementation of the proposed solutions of this project.

To achieve a high-level security for IoT devices, it is necessary to introduce a shared standard that can guarantee a common methodology for the design and development of the device, but above all, issues that are related to the updates.

6.3.1 Up-to-date Firmware

Once the system operative life is started, an important countermeasure to mitigate firmware tampering risk is keeping it always up to date to the latest manufacturer version. Firmware patch management for all the system and its critical infrastructure should be part of Standard Operative Procedures (SOP).

One problem of this countermeasure is that many firmware manufacturers do not release firmware patching often or do not release patches at all. System manufacturers should guarantee device firmware maintenance and updates for the entire system, assuring fast patching of potential vulnerabilities. There should be a continuous in-house firmware testing and verification for ensuring the absence of bugs; every manufacturer should guarantee this provision for at least the entire proposed guaranteed lifespan of the device.

On the other hand, however, updates can lead to vulnerability (vulnerable surfaces), potentially exploitable leaving a strong attack vector.

6.3.2 Firmware Threats and Exploitations

To understand how we can be protected from firmware tampering, we first have to understand all the ways a potential attacker has to compromise the integrity of a firmware platform.

The most insidious way to perpetuate an attack is by firmware image exploitation. Modern devices' OSes have the possibility to rewrite the platform firmware live; an attacker can modify the firmware image during its presence on the manufacturer device, during its download, or the writing process to the chip, for example, taking advantage of a machine with a compromised OS.

Regardless of the chosen exploitation vector, once reading/writing possibility of the firmware chip content is gained, an attacker can:

- Reverse-engineer the entire firmware, extract the file system, and understand how the entire device works, including knowing the possible use of known-to-be-vulnerable out-of-date API/libraries or unknown exploitable vulnerabilities in the firmware code itself;
- Insert a firmware backdoor, making the device covertly connected to a malicious Command & Control server;
- Change the device behavior, altering its performance, without altering the telemetry values reported to administrators;
- Find hard-coded private symmetric-cryptography keys/passwords/usernames or private certificates used to encrypt communications between the device and other systems; the attacker may be able to eavesdrop these private communications;
- Roll back the firmware to a previous legitimate version with known vulnerabilities he/she wants to exploit; this attack is particularly insidious because the pushed firmware is authentic, so it can easily survive most of the in-place controls, as usually they tend to check just the firmware source and/or the firmware integrity.

6.4 IPSE Dashboard and User Interfacing

In parallel with the above CHARIOT IPSE components, CHARIOT develops the Cognitive System and related methodology as the accompanying supervision, analytics, and prediction models, enabling high security and integrity of Industrial

IoT devices and systems. This includes the development of the intelligent behavior of the CHARIOT platform to enable IoT critical systems' secure interaction. In this direction, a web-of-things environment is being developed interacting with heterogeneous IoT devices and systems. This includes the development of the suitable interfaces for the topological and functional behavior models for the IoT system devices (components) as well as additional safety and privacy structures that communicate through open APIs and include security services employing Blockchain, IoT security profiles, and fog services.

This module will be mainly used to predict and avoid potentially endangering behavior in the IoT system that is now handled in an optimized way in cooperation with safety critical systems and run-time environments of the IoT system itself. This is using the existing IoT platform (IBM's Watson IoT) to demonstrate the CHARIOT innovative concept and capability and support integration with other safety, privacy, and machine-learning cloud services via relevant open APIs, thus supporting third-party integration and innovation.

At the same time, a security, privacy, and safety awareness configurable dashboard is being developed in order to remotely monitor in near real time the various IoT gateways and sensors in the CHARIOT IoT network. This will be the actual end-user interface to the CHARIOT solution and platform. The dashboard has been designed as part of the entire IPSE implementation and will also be used for both understanding of the IoT ecosystem topology as well as for the post data analytical purposes to assist in the refinement and improvements of PSS policies.

This advanced intelligence web-based dashboard utilizes the latest state-of-the-art web technologies in order to deliver rich content information to the LL users and achieve cross-browser and multi-device compatibility. The web development has been performed using the .NET Core framework and with the utilization of user interface implementation technologies like HTML, CSS, and JavaScript. Moreover, support on publish-subscribe-based messaging protocols as well as REST-API have also been utilized. Further to that, the dashboard follows rules regarding user friendliness, responsiveness, and configurability as web solution, based on the LLs needs. The Dashboard supports various components that are expected to support and enhance the user experience as well as the actual system operation and configuration such as interactive widgets, customizable entries, multiple components, data filtering, gauss controls and fusion charts, map, and top view visualization controls, etc. The following Figure presents a snapshot of this Dashboard for monitoring the devices in a train wagon.

As presented in the following (Figure 6.11), via the IPSE dashboard, the status of the IPSE Engines and fog node services can also be monitored. "Health visibility" (Figure 6.12) is a service which is checking the "live" status of each CHARIOT component in a graphical way.

Figure 6.11. IPSE dashboard—train wagon sensors visibility.

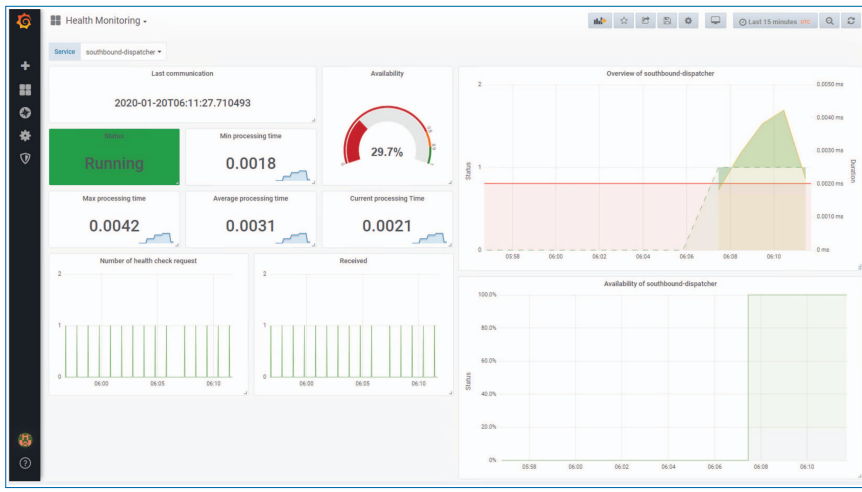


Figure 6.12. IPSE components health visibility.

Moreover, via IPSE dashboard, the end user is able to perform administrative and management activities related to:

- **Sensor management,** by securely adding or removing sensors on an existing industrial IoT topology.
- **Firmware management,** by securely updating the firmware of a sensor or a gateway and monitoring all checkpoints during this update process (via a workflow diagram) as presented in the following (Figure 6.13).

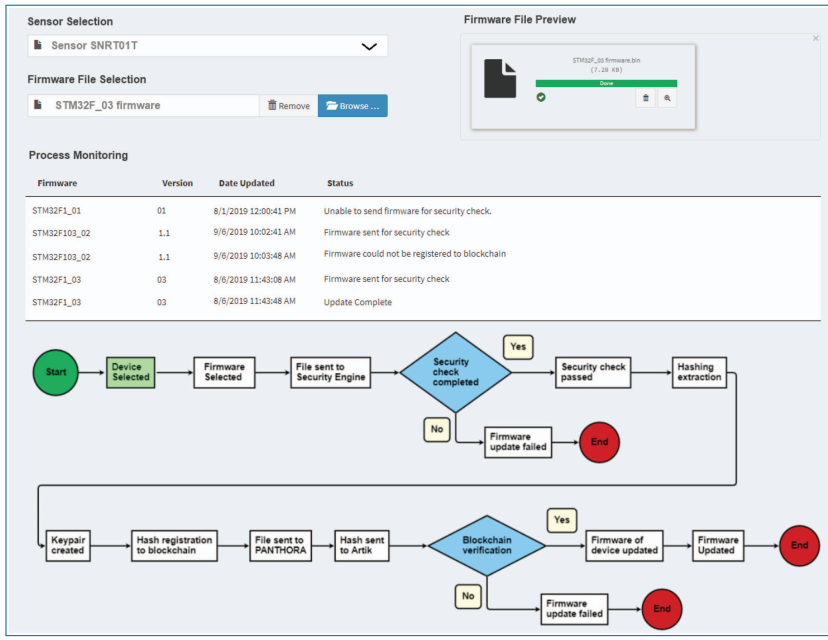


Figure 6.13. Firmware update and process visibility via the dashboard.

6.5 Conclusions and Future Work

Each engine of the CHARIOT IPSE provides several new innovations over the state-of-the-art in industrial IoT. Distributed safety supervision introduced by the safety supervision engine ensures fault-tolerant operation of the cognitive safety assurance components such as anomaly detectors and other online machine learning-based models. The relevant state is shared across multiple components in the Fog network to enable edge cloud orchestration by using the domain-specific language developed for IPSE. Having multiple active computation nodes in the industrial IoT network enables fault tolerance during safety hazards against partial network outages.

The Privacy engine ensures data privacy through encrypting data at the source, namely at the southbound dispatcher through a PKI supported by CHARIOT blockchain infrastructure. For industrial IoT setups with multiple contractors, our approach provides detailed privacy policies for every sensor data stream. On top of that, through the privacy-related constructs in the domain-specific language, privacy policies can be enforced even during partial network outage. Using CHARIOT Blockchain solution for handling PKI provides secure encryption for the multiple data streams handled by CHARIOT.

Finally, the security engine applies machine learning to identify security problems in possible firmware updates dynamically, and the signed firmware updates

are distributed across the industrial IoT setup. Deployment of the integrated IPSE provides CHARIOT living labs the capability of addressing their business-critical problems related to safety supervision, security, and privacy.

Acknowledgments

This project has received funding from the European Union's Horizon 2020 research and innovation program (No. 780075). The authors acknowledge the research outcomes of this publication belonging to the CHARIOT consortium.

References

[1] Ranjan, Rajiv, *et al.* "The Next Grand Challenges: Integrating the Internet of Things and Data Science." IEEE Cloud Computing 5.3 (2018): 12–26.
[2] Sfar, Arbia Riahi, *et al.* "A roadmap for security challenges in the Internet of Things." Digital Communications and Networks (2018): 118–137.
[3] Sanchez-Iborra, Ramon, and Maria-Dolores Cano. "State of the art in LP-WAN solutions for industrial IoT services." Sensors 16.5 (2016): 708.
[4] Pradhan, Dhiraj K., and Sudhakar M. Reddy. "A fault-tolerant communication architecture for distributed systems." IEEE transactions on Computers 9 (1982): 863–870.
[5] Ukil, Arijit, Soma Bandyopadhyay, and Arpan Pal. "Iot-privacy: To be private or not to be private." Computer Communications Workshops (INFOCOM WKSHPS), 2014 IEEE Conference on. IEEE, 2014.
[6] Shi, Weisong, *et al.* "Edge computing: Vision and challenges." IEEE Internet of Things Journal 3.5 (2016): 637–646.
[7] Sun, Xiang, and Nirwan Ansari. "EdgeIoT: Mobile edge computing for the Internet of Things." IEEE Communications Magazine 54.12 (2016): 22–29.
[8] Da Xu, Li, Wu He, and Shancang Li. "Internet of things in industries: A survey." IEEE Transactions on industrial informatics 10.4 (2014): 2233–2243.
[9] Porambage, Pawani, *et al.* "The quest for privacy in the internet of things." IEEE Cloud Computing 2 (2016): 36–45.
[10] Sadeghi, Ahmad-Reza, Christian Wachsmann, and Michael Waidner. "Security and privacy challenges in industrial internet of things." Design Automation Conference (DAC), 2015 52nd ACM/EDAC/IEEE. IEEE, 2015.
[11] Zhao, Yan Ling. "Research on data security technology in internet of things." Applied Mechanics and Materials. Vol. 433. Trans Tech Publications, 2013.

DOI: 10.1561/9781680836837.ch7

Chapter 7

Pattern-driven Security, Privacy, Dependability and Interoperability in IoT

By Nikolaos Petroulakis, Konstantinos Fysarakis, Henrich C. Pöhls, Vivek Kulkarni, George Spanoudakis, Arne Bröring, Manos Papoutsakis, Manolis Michalodimitrakis and Sotiris Ioannidis

This chapter presents the development of a pattern-driven approach for guaranteeing Security, Privacy, Dependability and Interoperability (SPDI) properties in the IoT domain. The chapter details how SPDI patterns can be introduced to guarantee multi-layer end-to-end properties, and how the enforcement of the patterns can help and satisfy the requirements for network dependability guarantees. Moreover, it briefly evaluates the presented approach and thus demonstrates how the application of patterns offers a solution to semantic interoperability challenges in IoT environments.

7.1 Introduction

While the fifth generation (5G) of mobile communications is already dawning upon us, the next steps in their evolution will be key in supporting this societal transformation, while also leading to a fourth industrial revolution that will impact multiple sectors. Next-generation networks, such as the Internet of Things (IoT), aim to create open and global networks for connecting smart objects, network elements, applications, web services, and end users. Research and industry

attempt to integrate this evolving technology and the exponential growth of IoT by overcoming significant hurdles such as dynamicity, scalability, heterogeneity, and end-to-end security and privacy. The introduction of digital technologies in economic and societal processes is key to addressing economic and societal challenges such as aging of population, ensuring societal cohesion, and sustainable development. IoT appears to be an important pillar of 5G. Global networks like IoT create enormous potential for new generations of IoT applications, by leveraging synergies arising through the convergence of consumer, business, and industrial internet, and creating open, global networks connecting people, data, and *things*. A series of innovations across the IoT landscape have converged to make IoT products, platforms, and devices technically and economically feasible. However, despite these advances, significant business and technical hurdles must be overcome before the IoT's potential can be realized.

Some important challenges and complexities include:

- Sustaining massively generated, ever-increasing, network traffic with heterogeneous requirements
- Adaptation of communication technologies for resource-constrained virtualized environments
- Provision of networking infrastructures featuring end-to-end connectivity, security, and resource self-configuration
- Trusted information sharing between tenants and host systems.

Overcoming these challenges requires the implementation and deployment stack of IoT applications. The overall aim of SEMIoTICS[1] proposes the development of a pattern-driven framework [1, 2], built upon existing IoT platforms, to enable and guarantee secure and dependable actuation and semi-autonomic behavior in IoT and Industrial Internet of Things (IIoT) applications. The SEMIoTICS framework supports cross-layer intelligent dynamic adaptation, including heterogeneous smart objects, networks, and clouds. To address the complexity and scalability needs within horizontal and vertical domains, SEMIoTICS develops and integrates smart programmable networking and semantic interoperability mechanisms.

The SEMIoTICS architectural framework (Figure 7.1) has been envisaged and developed for efficient interconnectivity of smart objects. Each layer contains specific developed modules able to handle different aspects and guarantee different properties. More specifically, Software Defined Networking (SDN) Orchestration layer provides data and control plane decoupling resulting in a cloud computing

1. SEMIoTICS: Smart End-to-end Massive IoT Interoperability, Connectivity and Security: http://www.semiotics-project.eu

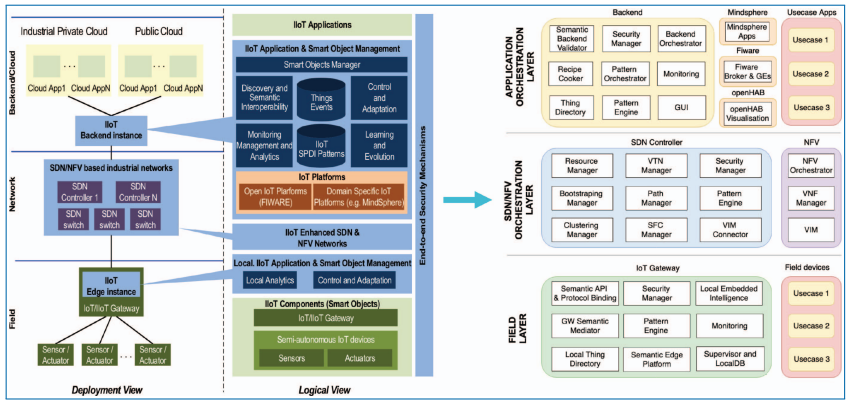


Figure 7.1. SEMIoTICS envisaged and developed architecture.

approach that facilitates network management and enables programmatically efficient network configuration meeting different IoT application requirements related to security, bandwidth, latency, and energy efficiency, using semantic information. Network Function Virtualization (NFV) Orchestration layer provides a flexible, programmable, dynamic, and scalable networking paradigm, making it ideal for satisfying the Quality of Service (QoS) demands of SEMIoTICS use cases. Field layer is responsible for hosting all types of IoT devices such as sensors and actuators as well as IoT gateway, which provides common way for communication and ensures enforcement of SPDI patterns in this layer. Finally, Application Orchestration layer consists of all applications receiving the communication from field layer.

This chapter is organized as follows: Section 7.2 presents the IoT challenges and the background of security, privacy, dependability and interoperability (SPDI)-related issues. The main concept and the key building blocks of the proposed pattern-driven IoT service orchestration solution are detailed in Section 7.3. Section 7.4 provides a glimpse on the proposed SPDI patterns. Finally, Section 7.5 features the concluding remarks and pointers to future work.

7.2 Background and Challenges

The background of the SPDI properties and the necessity for the usage of patterns to drive the fulfillment of the end users' goals in the directions of security, privacy, dependability and interoperability for the IoT are provided in this section. Furthermore, the IoT challenges stemming from each of the four domains are also briefly mentioned in order to highlight that being able to address them system-wide and in a *by-design-fashion* is of paramount importance in industrial or larger IoT environments.

7.2.1 IoT Security

Smart objects, IoT applications, and their enabling platforms are often vulnerable to security attacks and changing operating and context conditions that can compromise their security [3]. This is exacerbated by the fact that they typically generate, make use of, and inter-relate massive personal data in ways that can potentially breach legal and privacy requirements. Preserving security and privacy properties remains a particularly challenging problem, due to the difficulty of: (a) analyzing vulnerabilities in the complex end-to-end compositions of heterogeneous smart objects; (b) selecting appropriate controls for smart objects with heterogeneous resources/constraints (e.g., different schemes for ID and key management, different encryption mechanisms); and (c) preserving end-to-end security and privacy under dynamic changes in IoT applications and security incidents, in the context of the ever-evolving IoT threat landscape [4]. In this context, basic security tasks such as mutual authentication, encryption, and data integrity remain challenging in IoT, with a variety of lightweight solutions proposed [5]. Confidentiality and integrity protection mechanisms also require strong authentication and authorization mechanisms. Existing solutions requiring user involvement [6] or certificates [7] lack scalability and support for dynamic IIoT networks. Security and privacy at the IoT back-end, e.g., the cloud and centralized servers, requires distribution of data and processing, which necessitates the use of new distributed and/or collaborative paradigms of cloud computing [8]. To ensure that the most sensitive private data remain secure from source to end user and only accessible to authorized entities, one of the solutions is to encrypt data based on policies for access control, in addition to using secure communication channels. Thus, specific research areas such as Attribute-based Encryption (ABE) need to be improved in order to make them more efficient and scalable [9].

7.2.2 Privacy Invasion

With an increased consumer awareness by privacy invasions on general media or on industrial IoT applications, the need for privacy protection becomes immanent. From a business perspective, the problem of privacy comes framed as the problem to protect companies trade secrets. Additionally, privacy legislation can be a major concern in certain markets more than in others. Lately, the legal concerns have been fueled by heightened fines for privacy violations introduced in data protection laws, especially Europe's GDPR [10, 11]. The GDPR [10] requires among other things that personal data may be gathered only for a precisely specified purpose. It also requires to minimize the amount of collected data [12] and the data subject. Moreover, the individual person who's personal data are handled needs to give their informed consent a priori to the data gathering and must be able to intervene.

Further to being a legal or business challenge, privacy could also be seen as a human right in either way privacy matters in the IoT domain and as such we discuss some privacy challenges which have to be taken care of at the architectural level:

- **Handle IoT devices as first-class citizens:** Unsuspicious information communicated within the IoT could too easily be used by attackers to extract companies trade secrets or infringe the human right to protect personal information. Privacy requires to tackle the privacy problems already at the field level, i.e., already deploy sensors and actuators which are capable of tackling privacy problems [13]. As a minimum step, this requires to protect confidentiality of any data that is communicated, i.e., by encrypting traffic [14].
- **Minimize transferred data:** Data minimization can be achieved in many ways, by simple aggregation of data, but SEMIoTICS is even smarter and performs intelligent analytics locally [15].
- **Provision of privacy enhancing mechanisms and services** Required to implement in practical IoT deployments the technological advances that allow to provide data with certain quality guarantees while still containing less private information [16–18].
- **Specify privacy requirements:** A device that is communicating might leak, i.e., even encrypted traffic may still be vulnerable to being analyzed for length of packets, frequency of communication, and other observable communication meta-data. Thus, achieving unobservability of communication [19] would be the highest goal for IoT privacy [20]. To achieve privacy holistically and deploy the right mechanisms correctly, privacy must become part of the engineering framework [21].
- **Monitor potential privacy breaches:** The patterns approach allows to address and specify privacy requirements. But, the potential of design patterns for privacy goes beyond a single policy language cause; they allow communication between different actors in different domains. Finally, they are ideal for enabling information privacy into information systems [22].

The latter mentioned challenge of monitoring allows the data subject to exercise some form of oversight, which adds to the transparency of the data processing.

7.2.3 Network Dependability

Dependability is the ability of a system to deliver its intended level of service to its users [23]. The main attributes which constitute dependability are reliability, availability, safety, and maintainability. Dependable systems impose the necessity to provide higher fault and intrusion tolerance. The satisfaction of these attributes can avoid threats such as faults, errors, and failures offering fault prevention; fault

tolerance; and fault detection. More specifically, dependability in SEMIoTICS is focused on three major attributes such as reliability, availability, and fault tolerance as follows:

- **Reliability** is the ability of a system to perform a required function under stated conditions for a specified period of time [24]. It is an attribute of system dependability, and it is also correlated with availability. For hardware components, the property is usually provided by the manufacturer. This is calculated based on the complexity and the age of the component. Reliability can be classified into two main categories: the deterministic models and the probabilistic ones.
- **Availability** guarantees that information is available when it is needed [24]. The lack of availability in network transmissions has a severe influence on both the security and the dependability of network. More specifically, network availability is the ability of a system to be operational and accessible when required for use. Moreover, availability in networks is the probability of successful packet reception [25]. Other factors which affect the availability of a link are the transmission range of the signal strength, noise, fading effects, interference, modulation method, and frequency.
- **Fault Tolerance** is the ability of a system or component to continue normal operation despite the presence of hardware or software faults [24]. Network fault tolerance appears to be a critical topic for research [26]. The most common solutions to guarantee fault tolerance and avoid single point of failure include the replication of paths forwarding traffic in parallel, the use of redundant paths, and the ability to switch in case of failure (failover) and traffic diversity. Fault tolerance mechanisms exist in all layers of field, network, and back-end/cloud. In the field layer, failures involve the drop of sensors or actuators and the gateways. More specifically, fault tolerance in network architectures requires the design of a network able to guarantee avoidance of single or multiple link failures, faulty end hosts and switches, or attacks. The key technical solution of the problem includes the creation of a fault tolerance mechanism to provide open-flexible design where existing fault tolerance solutions are not effective.

Dependability analysis of an IoT system includes whether non-functional requirements such as availability, reliability, safety, and maintainability are preserved. The conditions depend on the respective dependability property that the system guarantee. The satisfiability of a property can be defined by a Boolean value (i.e., true, false), an arithmetic measure (i.e., delay), or probability measure (i.e., reliability/uptime availability).

7.2.4 IoT Interoperability

Interoperability gives an ability to a system or a product to connect and work with other systems or products. Interoperability is defined as *a characteristic of a product or system, whose interfaces are completely understood, to work with other products or systems, present or future, in either implementation or access, without any restrictions* [27]. The following types of interoperability can be distinguished and are covered in SEMIoTICS [28]:

- **Technological interoperability** enables seamless operation and cooperation on heterogeneous devices that utilize different communication protocols. Technological interoperability still remains a significant barrier in IoT settings as up to 60% of the overall potential value is currently locked due to lack of compatible solutions [29].
- **Syntactic interoperability** can establish clearly defined formats for data, interfaces, and encoding. In terms of syntactic interoperability, IoT vendors typically claim to utilize standardized and widely used technologies and platforms in order to increase the acceptance of their products. Nevertheless, the variety of common choices, such as the Constrained Application Protocol (CoAP), eXtensible Messaging and Presence Protocol (XMPP), Advanced Message Queuing Protocol (AMQP), MQ Telemetry Transport (MQTT), Devices Profile for Web Services (DPWS), and Universal Plug and Play (UPnP), leads to a fragmented landscape [30]. Mechanisms for resolving this issue include gateway proxies for the messaging protocols [31], but again their presence cannot be assumed in all environments and their introduction leads to increased complexity in interactions and additional costs.
- **Semantic interoperability** settles commonly agreed information models and ontologies for the used terms that are processed by the interfaces or are included in exchanged data. It is materialized by including information regarding the data (metadata) and linking each element to a commonly shared vocabulary. As IoT integrates an extremely large amount of heterogeneous entities, these entities need to be consistently and formally represented and managed. The Open Geospatial Consortium (OGC) [32] and Semantic Web Activity of the World Wide Web Consortium (W3C) [33] help provide enhanced descriptions and meaning to sensor data. Several semantic frameworks like Semantic Sensor Network (SSN) ontology [34] and IoT-Lite [35] have been proposed in the international literature. One of the main concerns leading to poor adoption of IoT semantic descriptions is that semantic techniques increase the complexity and processing time, and therefore, they are unsuitable for dynamic and responsive environments such as the IoT.

- **Organizational interoperability** cross-domain service integration and orchestration through common semantic and programming interfaces. The common interpretation of semantic information in a globally shared ontology could be useful in enabling cross-domain and cross-organization interactions. However, this is not always the case. Although several local systems may utilize popular or standardized ontologies, eventually they extend them and establish their own semantics and interfaces. The use of Semantic Information Broker (SIBs) is proposed the literature [36], while other approaches focus on providing common and generic Application Programming Interfaces (APIs) between the different IoT middleware platforms, towards a marketplace of applications and services (e.g., BIG IoT [37] project).

7.2.5 Achieving Security, Privacy, Dependability and Interoperability by Design

Research and industry communities alike stress the need to adopt a *by design* approach as the most effective means of addressing the above challenges, i.e., considering these early from the design phase. In this context, the pattern-driven approach of SEMIoTICS follows the *security-by-design* concept, which aims to guarantee system-wide security properties by virtue of the design of the involved systems and their subsystems. This is leveraged to provide orchestration-level SPDI guarantees, while encompassing all involved components and entities which are composed to create the orchestrations (e.g., physical devices and software). A key capability required in security-by-design is the ability to verify the desired security properties as part of the design process. A typical way to achieve this is using model-based techniques [38–40], whereby software component and service compositions are modeled using formal languages, and the required security properties are expressed as properties on the model [41]. The satisfiability of the required properties is based on model checking [42, 43]. Other approaches focus on software service workflows using business process modeling languages (e.g., Sec-MoSC [44]). Pino *et al.* [45] use Secure Service Orchestration (SSO) patterns to support the design of service workflows with required security properties, leveraging pattern-based analysis to verify security properties. This avoids full model checking that is computationally expensive and non-scalable to larger systems, such as the IoT. Moreover, some model-based approaches (e.g., [45]) support the transformation of security requirements to code for automated checking of the required properties, both at design and at run time.

The SEMIoTICS pattern-driven framework's operation is inspired by the many works that have successfully used design patterns as a mean to communicate across

different stakeholder domains and to reuse proven solutions for problems which occur over and over again. The initial works date back to the 1977 book "A Pattern Language: Towns, Buildings, Construction" by Alexander *et al.* [46], where the concept of reusable design solutions for architectural problems was introduced. The pattern approach has ever since been successfully applied in other domains, e.g., software design [47], security [48], privacy [49]. Here, SEMIoTICS, especially, builds upon ideas from similar pattern-based approaches used in service-oriented systems [50, 51], cyber-physical systems [52], and networks [53, 54], covering more aspects in addition to Security, and especially privacy [55, 56] and also providing guarantees and verification capabilities that span both the service orchestration and deployment perspectives, as detailed in Section 7.3 above.

7.3 SPDI Patterns

To enable the pattern-driven approach, it is necessary to develop a language for specifying the components that constitute IoT applications along with their interfaces and interactions. In this context, the definition of the various functional and non-functional properties of IoT components and their orchestrations is required in the form of a model. The defined model appears in Figure 7.2 and is presented in detail in [2]. A model with such characteristics effectively serves as a general "architecture and workflow model" of the IoT application.

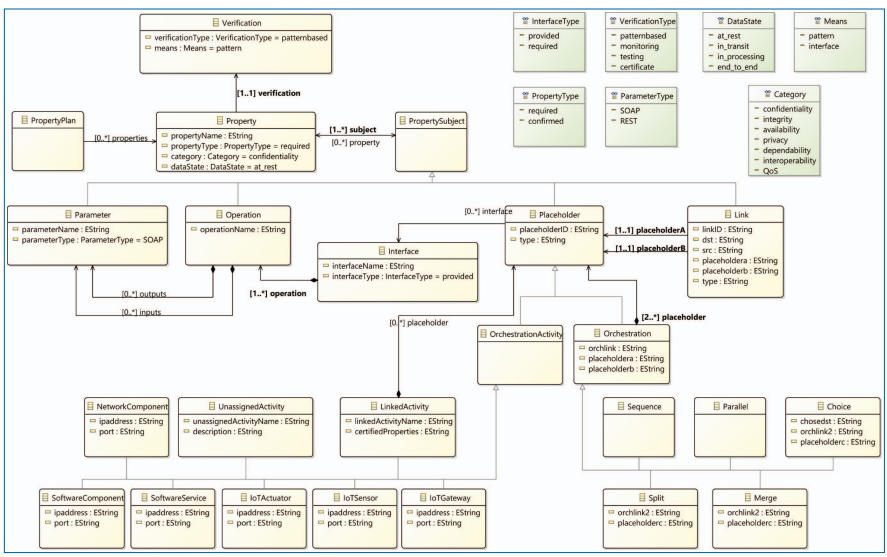


Figure 7.2. IoT orchestrations system model.

7.3.1 Pattern Language

Once defined, this model is used to derive a language which will allow the definition of pattern rules and facts which, consequently, enable the reasoning required for verifying SPDI and QoS properties in specific IoT applications and subsequently enable different types of adaptations. The derived language for defining IoT application models adopts an orchestration-based approach. An orchestration of activities may be of different types depending on the order in which the different activities involved in it must be executed (e.g., sequence, parallel, choice, merge). Moreover, an orchestration involves orchestration activities. The implementation of an activity in an IoT application orchestration may be provided by a software component, software service, network component, an IoT sensor, actuator or gateway, as well as a sub-orchestration of IoT application activities of the previous types. These types of IoT application activity implementers are grouped under the general concept of a placeholder, which is accessible through a set of interfaces.

Overall, this language: (i) provides constructs for expressing and encoding dependencies between SPDI properties at the component and at the composition/orchestration level; (ii) is structural, without prescribing exactly how the functions should be executed nor, e.g., how the ports ensure communication; (iii) supports the static and dynamic verification of SPDI properties, and; (iv) can be automatically processable by the SEMIoTICS framework so that IoT applications can be adapted at run time.

Patterns expressed in the above-defined language enable the pattern-based IoT application management process followed in SEMIoTICS, in which patterns are used to: design IoT applications that satisfy required SPDI properties; verify that existing IoT applications satisfy required SPDI properties at design time, prior to the deployment of the application; and enable the adaptation of IoT applications or partial orchestrations of components within them at run time in a manner that guarantees the satisfaction of required SPDI properties.

To fulfill the above, SPDI patterns encode proven dependencies between security, privacy, dependability and interoperability properties of individual components of IoT applications and corresponding properties of orchestrations of such components. More specifically, a pattern encodes relationships of the form:

$$P1 \wedge P2 \wedge \ldots \wedge Pn \rightarrow Pn+1 \tag{7.1}$$

In the above, Pi (i = 1,…,n) are properties of individual components, and Pn+1 is a property of the orchestration of these components. The relation encoded by a pattern is an entailment relation.

The run-time adaptations that can be enabled by SPDI patterns may take three forms: (a) to replace particular components of an orchestration; (b) to change the structure of an orchestration; and (c) a combination of (a) and (b).

7.3.2 Machine-processable Pattern Encoding

An important requirement for implementing the SPDI pattern-driven management and adaptation of the IoT infrastructure is to support the automated processing of developed patterns. To achieve this, the SPDI patterns can be expressed as *Drools* [57] business production rules, and the associated rule engine, by applying and extending the Rete algorithm [58]. The latter is an efficient pattern-matching algorithm known to scale well for large numbers of rules and datasets of facts, thus allowing for an efficient implementation of the pattern-based reasoning process. A Drools production rule has the following generic structure:

```
rule name
<attributes>*
when <conditional element>*
then <action>*
end
```

7.3.3 Reasoning Components

Pattern-related components are present in all layers of the SEMIoTICS framework (Figure 7.3), in line and towards realizing the SEMIoTICS vision of embedded intelligence across all layers of the IoT deployment.

In more detail, these components are:

- *Pattern Orchestrator*: Module featuring an underlying semantic reasoner able to understand IoT Orchestrations and workflows, as received from the Recipe Cooker module, and transform them into composition structures (orchestrations) to be used by architectural patterns to guarantee the required properties. The Pattern Orchestrator is then responsible to pass said patterns to the corresponding Pattern Engine (as defined in the back-end, network and field layers), selecting for each of them the subset of these that refer to components under their control (e.g. passing network-specific patterns to the Pattern Engine present in the SDN controller).
- *Backend Pattern Engine*: Features the pattern engine for the SEMIoTICS back-end, along with associated subcomponents (knowledge base, reasoning engine). It is able reason on the SPDI properties of aspects pertaining to the operation of the SEMIoTICS back-end. Moreover, at run time the

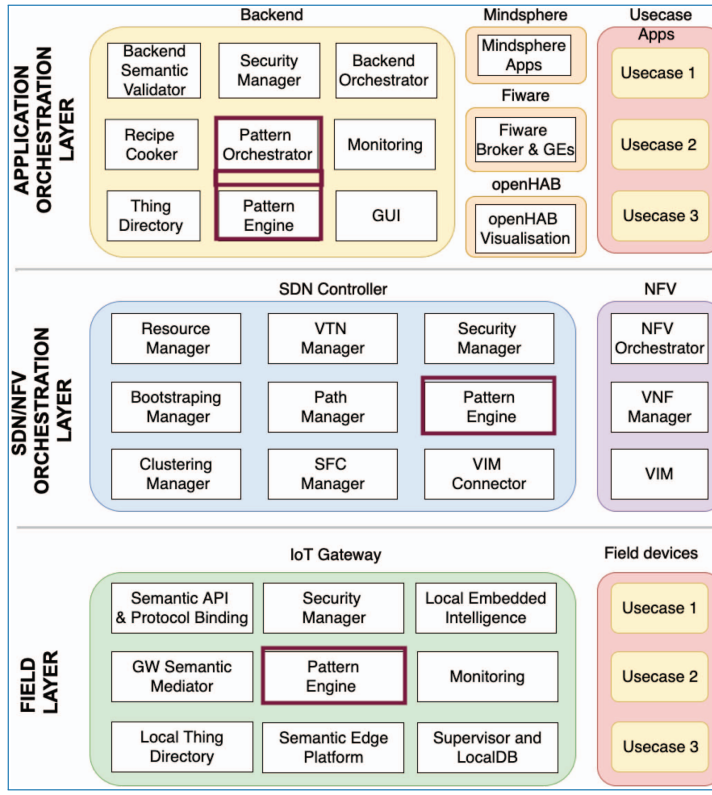


Figure 7.3. Pattern reasoning engines at all layers of the SEMIoTICS architecture.

Backend Pattern Engine may receive fact updates from the individual Pattern Engines present at the lower layers (Network & Field), allowing it to have an up-to-date view of the SPDI state of said layers and the corresponding components

- *Network Pattern Engine*: Integrated in the SDN controller to enable the capability to insert, modify, execute, and retract network-level patterns at design or at run time. It is supported by the integration of all required dependencies within the network controller, as well as the interfaces allowing entities that interact with the controller to be managed based on SPDI patterns at design and at run time.
- *Field Layer Pattern Engine*: Typically deployed on the IoT/IIoT gateway, able to host design patterns as provided by the Pattern Orchestrator. Since the compute capabilities of the gateway can be limited, the module is able to host patterns in an executable form compared to the pattern rules as provided in the other layers.

7.4 Pattern Enforcement's and Evaluation in SEMIoTICS Use Cases

Patterns in SEMIoTICS framework are enforced to satisfy SPDI properties on the different use cases. In the next subsections, a demonstrated scenario to enable pattern-driven IoT orchestration to satisfy QoS requirements is presented. Moreover, the policy enforcement through the pattern engine to monitor privacy violations is provided. Finally, the service function chaining patterns are also described to enable the classification and forward traffic through the respective security service functions in the SEMIoTICS pattern framework.

7.4.1 Pattern-enabled IoT Orchestrations

A demonstration scenario that relies on the SEMIoTICS pattern-driven network interface and its capabilities was designed and developed in industrial IoT environments and more specifically oil leakage detection in wind turbines through video monitoring. The overarching aim of the scenario is to distribute a complex application (composed of multiple tasks) to a network of IoT device and specify constraints (through patterns) on the network orchestration. In this context, the developed scenario leverages a user-friendly design and deployment of IoT orchestrations. The two key research innovation of the scenario and associated demonstration relate to: (1) True distribution of application flows over multiple devices and representing the network perspective and (2) Automated enforcement of network orchestration constraints by defining them as SEMIoTICS patterns.

In the above, other than the user-friendly, graphical interface and distributed nature of defining the IoT orchestrations involved (including where/on which devices parts of a flow are deployed), we also want to define SPDI and QoS between these deployments. Focusing on the network aspects, while maintaining the high-level abstractions needed for user-friendliness, a "Network Link" node enables direct communication between distributed Node-RED instances. Said "Network Link" node enables definition of QoS constraints (e.g., minimum bandwidth, latency) and the whole orchestration specification (a "Recipe") [59, 60], and the QoS constraints are translated into the SEMIoTICS pattern language and sent to Pattern Orchestrator [61, 62]. From the latter, the information is relayed to the network (SDN) Pattern Engine. A high-level view of this process is shown in Figure 7.4.

7.4.2 Security and Privacy Policy Enforcement Patterns

The main storyline is focused on a patient living in a smart home environment. In this environment, the patient's information is kept confidential by default;

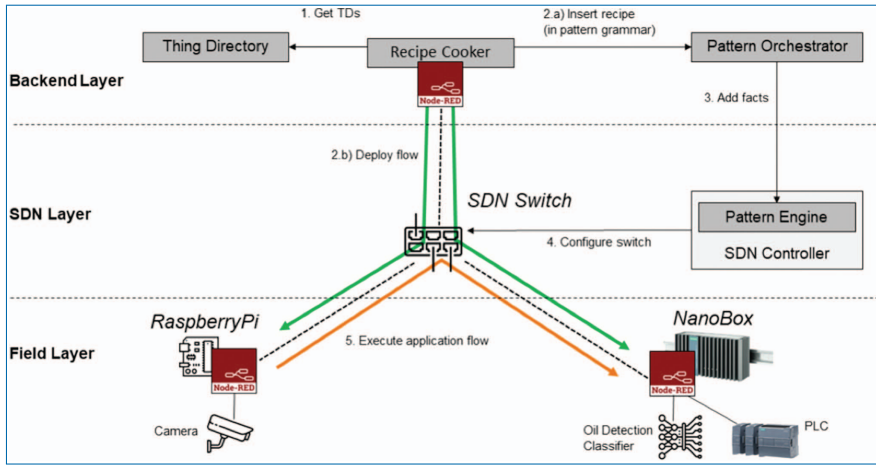


Figure 7.4. Pattern enforcement QoS [61].

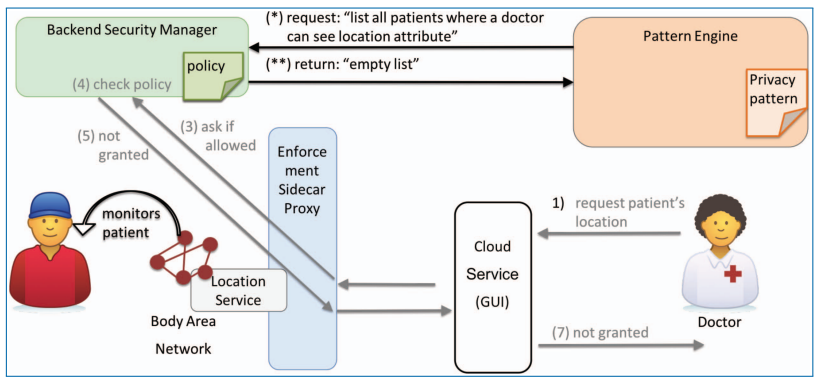


Figure 7.5. Interaction security manager and pattern engine.

however, in the event of an emergency, e.g., a patient fall, this information needs to be available to a selected set of authorized users, e.g., in this case medical personnel. The process, also depicted in Figure 7.5, includes a doctor's initial request to a service where the request is checked for validity by a *policy enforcement point* (PEP). Asking a SEMIoTICS component called *Security Manager* is only needed if the service would be exposed by the field level and after the synchronization and evaluation of the Security Manager. Based on this information, the PEP will then decide if the request or the response will be granted or not.

With respect to the scope of privacy, the SEMIoTICS *Security Manager* as depicted in Figures 7.3 and 7.5 acts to ensure that the privacy properties imposed by the pattern are technically enforced. In detail, this happens by enabling the interaction of a Security Manager located in the back-end with the Pattern Engine through the associated privacy pattern rule as depicted in Figure 7.5. The goal is that the

Pattern Engine is able to check if the current policy used to make the decisions inside the Backend Security Manager is conforming to the SPDI Patterns as specified by the application. The Pattern Engine thus periodically makes requests to the Security Manager to obtain the necessary information to judge the quality of the currently enforced policy. For example, to check that no one has access to a patient's location, it queries to obtain a list of entity that would be granted access. Based on the response, the appropriate pattern rule expressing the SPDI pattern can reason on the answers received from the Security Manager. For example, under normal health conditions of all patients that response shall be an empty list. By reasoning over the response, the Pattern Engine is able to identify if the policy being enforced in SEMIoTICS is compliant with the privacy pattern; this is done without having to check the actual technical enforcement (e.g., the access control policy as specified in the PEP), but rather by checking its effectiveness; further, it is able to reflect this to the outside via an API call that will allow other components to retrieve the SPDI status.

7.4.3 Service Function Chaining and SPDI Patterns

E-health monitoring systems situated at homes can facilitate the monitoring of patients' activities and enable the remote provision of healthcare services. They improve the quality of elder population well-being in a non-obtrusive way, allowing greater independence, maintaining good health, preventing social isolation for individuals, and delay their placement in institutions such as nursing homes and hospitals. One of the scopes of SEMIoTICS is to provide security guarantees through the traffic forwarding via different network security functions by applying the Service Function Chaining (SFC). The main focus of scenario is to support the traffic classification based on the predefined SFC for providing secure chains to forward the different kind of traffic of this use case. SEMIoTICS framework can be applied in order to support the SFC mechanism to guarantee security and dependability based on the defined SPDI patterns instantiating the required (i) Virtual Network Function (VNFs) and (ii) SFC for assuring the SPDI requirements. Traffic classification is based on the predefined SFC for providing secure chains to forward the different kinds of traffic of this use case. The procedure of instantiation and the identification of the respective SFCs and the VNFs based on the patterns are depicted in the Figure 7.6. Considering the different types of traffic reaching the back-end where the chaining of services will take place, a variety of intricacies can be observed such as of low trust and low priority, low bandwidth and latency, medium trust but high priority, medium trust and of low priority, and finally high trust and high priority, as low latency and relatively high bandwidth.

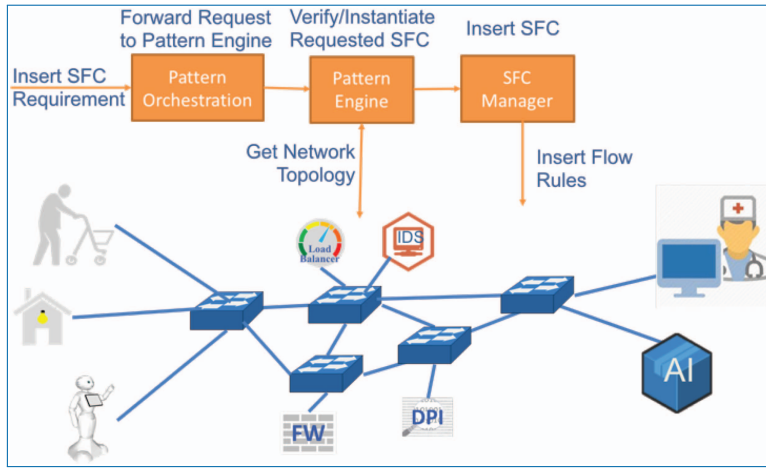


Figure 7.6. Pattern enforcement service function chains.

The design of an efficient control flow mechanism is required to be used not only to verify SFC and VNFs but also to instantiate them for assuring the SPDI requirements based on the enforcement of the respective SPDI patterns. When an SFC cannot be verified, the required VNFs are requested by the Virtual Infrastructure Manager (VIM) via NFV Orchestrator to identify them or to instantiate them if they do not exist. The procedure of instantiation and the identification of the respective SFCs and the VNFs based on the patterns are depicted in the Figure 7.6 including the following interactions with the components of the SEMIoTICS architecture. Pattern Orchestrator forwards a specific chain request to the pattern engine for forwarding the traffic between entities through a specific chain of functions. Pattern engine forwards this request to the SFC manager which is located in the SDN controller responding to the pattern engine whether the chain exist or not. If the chain exists, then a respond of the chain satisfaction is returned to the Pattern Orchestrator. If the chain does not exist, then a request is forwarded to the VIM asking whether the service functions exist or not. If functions exist in the VIM, then the chain can be instantiated in the SFC Manager and a respond of the chain satisfaction is returned to the Pattern Orchestrator. If functions do not exist in the VIM, then a function instantiation request is forwarded to the NFV Orchestrator, which is responsible to instantiate them in the VIM. Then, the chain can be instantiated in the SFC Manager, and a respond of the chain satisfaction is returned to the Pattern Orchestrator. Based on the provided dynamic instantiation of service chains and service functions through the pattern engine, the potentiality within this use case can be increased and extended by the support of additional service chains to enable traffic classification through different combinations of service functions to guarantee different secure end-to-end traffic forwarding.

7.5 Conclusion

This chapter presented a pattern-driven framework addressing the often very diverse and complex and not well-scaling requirements of commercial, societal, and industrial IoT applications. The work presented the SEMIoTICS framework's approach towards the development of patterns for orchestration of smart objects and IoT platform enablers in IoT applications with guaranteed security, privacy, dependability and interoperability properties. The definition of the pattern language and its development was also presented. Moreover, the chapter showcased the usefulness of a pattern-driven approach by describing how it works to increase the security, privacy, interoperability and dependability in very different and specific application scenarios. With this approach, the very diverse and system-spanning goals of even large-scale IoT deployments also for industrial IoT can be defined and securely and reliably orchestrated to meet the required SPDI properties.

Acknowledgments

This work has received funding from the European Union's Horizon 2020 research and innovation program under grant agreements No. 780315 (SEMIoTICS).

References

[1] N. E. Petroulakis *et al.*, "SEMIoTICS Architectural Framework: End-to-end Security, Connectivity and Interoperability for Industrial IoT." In 2019 IEEE Global IoT Summit (GIoTS), 2019.

[2] K. Fysarakis *et al.*, "Architectural Patterns for Secure IoT Orchestrations." In 2019 IEEE Global IoT Summit (GIoTS), 2019.

[3] M. Kert *et al.*, "State of the Art of Secure ICT Landscape." NIS Platform WG 3, V2, April 2015.

[4] ENISA, "Threat Landscape Report." https://www.enisa.europa.eu/publications/enisa-threat-landscape-report-2016, 2016.

[5] C. Manifavas, G. Hatzivasilis, K. Fysarakis, K. Rantos, "Lightweight Cryptography for Embedded Systems – A Comparative Analysis." In Data Privacy Management and Autonomous Spontaneous Security. DPM 2013, SETOP 2013. Lecture Notes in Computer Science, vol 8247. Springer, Berlin, Heidelberg, 2014.

[6] T. Marktscheffel, W. Gottschlich, W. Popp, P. Werli, S. D. Fink, A. Bilzhause and H. de Meer. "QR Code Based Mutual Authentication Protocol for Internet of Things." In Proc. of The 5th workshop on IoT-SoS: Internet of Things Smart Objects and Services (WOWMOM SOS-IOT 2016), IEEE, 2016.

[7] R. Hummen *et al.*, "Towards viable certificate-based authentication for the internet of things." In Proceedings of the 2nd ACM workshop on Hot topics on wireless network security and privacy, 2013.

[8] Big DataWorking Group. Expanded Top Ten Big Data Security and Privacy Challenges. Cloud Security Alliance, 2013.

[9] M. Chase and S. S. Chow. Improving privacy and security in multi-authority attribute-based encryption. Proceedings of the 16th ACM conference on Computer and communications security – CCS'09, page 121, 2009.

[10] European Parliament and the Council of the European Union, "Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation)." In Official Journal L 119 of 4.5.2016, pp. 1–88, 2016.

[11] J. Leyden, "Last year's ICO fines would be 79 times higher under GDPR." The Register, http://www.theregister.co.uk/2017/04/28/ico_fines_post_gdpr_analysis/, 2017.

[12] EU Article 29 Data Protection Working Party, "Opinion 8/2014 on the on Recent Developments on the Internet of Things." 2014.

[13] H. C. Pöhls *et al.*, "RERUM: Building a Reliable IoT upon Privacy- and Security- enabled Smart Objects." In Proc. of the Workshop on Internet of Things Communications and Technologies (IEEE WCNC 2014), pp. 122–127. IEEE, 2014.

[14] Z. Tragos, V. Angelakis, A. Fragkiadakis, D. Gundlegard, S. Nechifor, G. Oikonomou, H. C. Pöhls and A. Gavras, "Enabling reliable and secure IoT-based smart city applications." In Proc. of Conference on Pervasive Computing and Communications (IEEE PERCOM 2014), pages 111–116. IEEE, 2014.

[15] SEMIoTICS, "Deliverable D4.3 – Embedded Intelligence and Local Analytics." https://ec.europa.eu/research/participants/documents/downloadPublic?documentIds=080166e5c9a45e55, 2019.

[16] H. C. Pöhls and M. Karwe, "Redactable Signatures to Control the Maximum Noise for Differential Privacy in the Smart Grid." In Proc. of the 2nd Workshop on Smart Grid Security (SmartGridSec 2014), Springer, 2014.

[17] T. Lorünser, D. Slamanig, T. Länger and H. C. Pöhls. "PRISMACLOUD Tools: A Cryptographic Toolbox for Increasing Security in Cloud Services." In Proc. of the Workshop on Security, Privacy, and Identity Management in the Cloud to be held at the 11th International Conference on Availability, Reliability and Security (ARES SECPID 2016), Conference Publishing Services (CPS), August, 2016.

[18] D. von Oheimb and J. Cuellar, "Designing and Verifying Core Protocols for Location Privacy." In Proc. of International Conference on Information Security (ISC), pp. 502–516, Springer, 2006.

[19] R. C. Staudemeyer, H. C. Pöhls and M. Wojcik. "The road to privacy in IoT: beyond encryption and signatures, towards unobservable communication." In Proc. WOWMOM 2018, IEEE, July, 2018.

[20] R. C. Staudemeyer, H. C. Pöhls and M. Wojcik. "What it takes to boost Internet of Things privacy beyond encryption with unobservable communication: a survey and lessons learned from the first implementation of DC-net." In Journal of Reliable Intelligent Environments (JRIE), 5 (1), pp. 41–64, 2019.

[21] A. Kung *et al.* "A Privacy Engineering Framework for the Internet of Things." In Proc. of International Conference on Computers, Privacy and Data Protection 2016 (CDPD 2016), volume 36 of LGTS. Springer, 2016.

[22] N. Doty, M. Gupta, "Privacy design patterns and anti-patterns." In: Workshop "A Turn for the Worse: Trustbusters for User Interfaces Workshop" at SOUPS 2013, 2013.

[23] J. Laprie, "Dependable computing and fault-tolerance." in Digest of Papers FTCS-15, 1985.

[24] A. Geraci, F. Katki, L. McMonegal, B. Meyer, J. Lane, P. Wilson, J. Radatz, M. Yee, H. Porteous and F. Springsteel, "IEEE standard computer dictionary: Compilation of IEEE standard computer glossaries." 1991.

[25] PP. Park, P. Di Marco, C. Fischione and K. Johansson, "Modeling and optimization of the IEEE 802.15. 4 protocol for reliable and timely communications." Parallel and Distributed Systems, vol. 24, 2013.

[26] J. Chen, J. Chen, F. Xu, M. Yin and W. Zhang, "When Software Defined Networks Meet Fault Tolerance: A Survey." Springer International Publishing, pp. 351–368, 2015.

[27] G. Aful, "Definition: Interoperability." Available: http://interoperability-definition.info/en/, 2018.

[28] G. Hatzivasilis, I. Askoxylakis, G. Alexandris, D. Anicic, A. Bröring, V. Kulkarni, K. Fysarakis and G. Spanoudakis, "The Interoperability of Things: Interoperable solutions as an enabler for IoT and Web 3.0." IEEE International Workshop on Computer-Aided Modeling Analysis and Design of Communication Links and Networks 2018 (IEEE CAMAD 2018), Barcelona, Spain, Sept. 17–19, 2018

[29] J. Manyika *et al.*, 2015. Unlocking the potential of the Internet of Things. McKinsey Global Institute Report, McKinsey & Company, June 2015, pp. 1–4, 2015.

[30] A. Al-Fuqaha *et al.*, 2015. Internet of Things: a survey on enabling technologies, protocols, and applications, IEEE Communication Surveys & Tutorials, IEEE, vol. 17, issue 4, pp. 2347–2376.

[31] E. Palavras, K. Fysarakis, I. Papaefstathiou and I. Askoxylakis, "SeMIBIoT: Secure Multi-Protocol Integration Bridge for the IoT." 2018 IEEE International Conference on Communications (ICC), pp. 1–7, 2018.

[32] http://www.opengeospatial.org/

[33] https://www.w3.org/2001/sw/

[34] "The SSN ontology of the W3C semantic sensor network incubator group." Web semantics: science, services and agents on the World Wide Web 17 (2012): 25–32.

[35] M. Bermudez-Edo *et al.*, "IoT-Lite: a lightweight semantic model for the Internet of Things." Intl. IEEE Conferences, Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld), 2016.

[36] J. Kiljander *et al.*, Semantic interoperability architecture for pervasive computing and Internet of Things. IEEE Access, IEEE, vol. 2, pp. 856–873, 2014.

[37] A. Bröring *et al.*, Enabling IoT ecosystems through platform interoperability. IEEE Software, IEEE, vol. 34, issue 1, pp. 54–61, 2017.

[38] M. Bartoletti *et al.*, "Semantics-based design for secure web services." IEEE Trans. on Software Engineering, 2008.

[39] M. Deubler *et al.*, "Sound development of secure service-based systems." In Proc. of the 2nd Int. Conf. on Service oriented computing. ACM, 2004.

[40] G. Geor *et al.*, "Verification and trade-off analysis of security properties in UML system models." IEEE Trans. on Software Engineering, 36(3): 338–356, 2010.

[41] J. Dong *et al.*, "Automated verification of security pattern compositions." Inf. Softw. Technol., vol. 52, no. 3, 2010.
[42] I. Siveroni, A. Zisman and G. Spanoudakis, "A UML-Based Static Verification Framework for Security, Requirements." Engineering Journal, 15(1): 95–118, 2010.
[43] S. Rossi, "Model checking adaptive multilevel service compositions." International Workshop of Formal Aspects of Component Software, 2010.
[44] A. R. Souza *et al.*, "Incorporating Security Requirements into Service Composition: From Modelling to Execution." In ICSOC-ServiceWave'09, 2009.
[45] L. Pino, K. Mahbub and G. Spanoudakis, "Designing Secure Service Workflows in BPEL." International Conference on Service-Oriented Computing, 2014.
[46] C. Alexander, S. Ishikawa and M. Silverstein, "A Pattern Language: Towns, Buildings, Construction." Oxford University Press, Oxford, 1977.
[47] E. Gamma, R. Helm, R. Johnson and J. Vlissides, "Design Patterns: Elements of Reusable Object-Oriented Software." Addison-Wesley, Boston, 1994.
[48] M. Schumacher *et al.*, "Security Patterns – Integrating Security and Systems Engineering." Wiley, West Sussex, 2006.
[49] M. Hafiz, "A collection of privacy design patterns." In: Proceedings of the 2006 Conference on Pattern Languages of Programs, PLoP 2006, pp. 7:1–7:13. ACM, New York, 2006.
[50] L. Pino *et al.*, "Discovering Secure Service Compositions." 4th International Conference on Cloud Computing and Services Sciences (CLOSER 2014), Barcelona, Spain, 2014.
[51] L. Pino *et al.*, "Pattern Based Design and Verification of Secure Service Compositions." IEEE Trans. on Services Computing, 2017.
[52] A. Maña *et al.*, "Extensions to Pattern Formats for Cyber Physical Systems." Proceedings of the 31st Conference on Pattern Languages of Programs (PLoP'14), 2014.
[53] N. Petroulakis *et al.*, "Patterns for the design of secure and dependable software defined networks." Computer Networks 109 (2016): 39–49, 2016.
[54] N. Petroulakis *et al.*, "Fault Tolerance Using an SDN Pattern Framework." 2017 IEEE Global Communications Conference (GLOBECOM), 2017.
[55] T. Lorünser *et al.*, "Towards a new paradigm for privacy and security in cloud services." In: CSP Forum 2015. CCIS, vol. 530, pp. 14–25. Springer, Heidelberg, 2015.

[56] T. Länger, H. C. Pöhls and S. Ghernaouti, "Selected Cloud Security Patterns to Improve End User Security and Privacy in Public Clouds." In Proc. of Privacy Technologies and Policy – 4th Annual Privacy Forum (APF 2016), Springer, 2016.
[57] Business Rules Management System (BRMS), https://www.drools.org
[58] C. L. Forgy, "Rete: A fast algorithm for the many pattern/many object pattern match problem." Artif. Intell., vol. 19, no. 1, pp. 17–37, Sep. 1982.
[59] J. Seeger, R. A. Deshmukh, V. Sarafov and A. Bröring, "Dynamic IoT Choreographies – Managing Discovery," Distribution, Failure and Reconfiguration. IEEE Pervasive Computing, 18(1), pp. 19–27, 2019.
[60] A. S. Thuluva, A. Bröring, G. P. Medagoda Hettige Don, D. Anicic and J. Seeger, "Recipes for IoT Applications." Proceedings of the 7th International Conference on the Internet of Things (IoT 2017), 22–25. October 2017, Linz, Austria. ACM, 2017.
[61] A. Bröring, J. Seeger, M. Papoutsakis, K. Fysarakis and A. Caracalli, "Networking-Aware IoT Application Development." Sensors 2020, 20(3).
[62] J. Seeger, A. Bröring, M.-O. Pahl and E. Sakic, "Rule-Based Translation of Application-Level QoS Constraints into SDN Configurations for the IoT." EuCNC 2019, 18–21. June, Valencia, Spain. IEEE, 2019.

DOI: 10.1561/9781680836837.ch8

Chapter 8

Enabling Continuous Privacy Risk Management in IoT Systems

By Victor Muntés-Mulero, Jacek Dominiak, Elena González and David Sanchez-Charles

8.1 Introduction

The next-generation IoT systems need to perform distributed processing and coordinated behavior across IoT, edge, and cloud infrastructures; manage the closed loop from sensing to actuation: and cope with vast heterogeneity, scalability, and dynamicity of IoT systems and their environments (Ferry *et al.*, 2018). To unleash the full potential of IoT, it is essential to facilitate the creation and operation of trustworthy Smart IoT Systems or, for short, TSIS. TSIS typically operate in changing and often unpredictable environments. Thus, the ability of these systems to continuously evolve and adapt to their new environment is essential to ensure and increase their trustworthiness, quality, and user experience. Besides, by 2021, the number of "connected things" will grow to 25 billion, according to Gartner.[1] Thus, processes that were formerly run by humans will be automated, making it much more difficult to control data ownership, privacy, and regulatory compliance.

Yan *et al.* (2014) described the different dimensions of trust for IoT systems, concluding that risk management is essential to guarantee trustworthiness.

1. https://www.gartner.com/en/newsroom/press-releases/2018-11-07-gartner-identifies-top-10-strategic-iot-technologies-and-trends

Poor risk management together with a reactive strategy usually forces companies to continuously re-factor application architectures to improve software quality and security, incurring high re-implementation costs (Boehm and Turner, 2003). In general, there is a lack of solutions to support continuous control of risks through evidence collection. Companies have little control on actual effectiveness of the mitigation actions defined during risk management processes. Besides, many companies fill this gap by using manual procedures based on storing all the information in spreadsheets, by departments and locally (Ahmad *et al.*, 2012). This approach rapidly turns inefficient as projects or teams grow.

In parallel, GDPR discusses data protection by design and by default, remarking that it is essential to consider privacy from the beginning to address related issues successfully. This is especially true in the IoT arena, where technologies are not consolidated yet and mixing legal requirements with a deep technical understanding is challenging. In particular, understanding privacy-related vulnerabilities and including privacy considerations in a continuous risk management process is difficult. On the one hand, knowledge related to vulnerabilities connected to privacy issues is not so commonplace. On the other hand, continuous evidence collection to support risk management is usually key, but most monitoring approaches focus on collecting evidences from the TSIS infrastructure or technical architectural components. However, privacy-related risks are usually detected by analyzing functional descriptions of the system (Wuyts, 2015) (e.g., data flows). Connecting this functional level with the components of the architecture that are being monitored is not trivial. Recognizing the overlap between privacy and security is key to determining when existing security risk models may be applied to address privacy concerns (Brooks *et al.*, 2017).

In this chapter, we establish the basis to create an Automated Vulnerability Detector (AVD) for early detection of privacy issues. We also improve continuous risk management in TSIS development by embedding privacy-related risks explicitly through the combined use of models for both the architecture and the data flow implemented on the components of the architecture. We present a new approach that, through linking GeneSIS (Ferry *et al.*, 2019) models and Data Flow Diagrams (DFDs) (DeMarco, 1979), improves risk assessment process for privacy-related risks. We achieve this by enabling the use of the information that is typically collected from the infrastructure to control security, thanks to the link that LINDDUN (Wuyts, 2015) establishes between privacy and security threads in STRIDE (Shostack, 2014).

This chapter is organized as follows. Section 8.2 provides an analysis of the state of the art. Section 8.3 presents our methodology. Then, Section 8.4 describes the basic steps to create an AVD taking LINDDUN as the baseline reference. In Section 8.5, we describe the mapping between the architecture model and DFDs as well as our risk model. Finally, in Section 8.6, we draw some conclusions.

8.2 State of the Art

There exist quantitative risk methodologies and tools, like RiskWatch[2] or ISRAM (Karabacak and Sogukpinar, 2005) and qualitative risk methodologies such as OCTAVE (Alberts and Dorofee, 2002), CORAS (Lund *et al.*, 2010), or STRIDE (Shostack, 2014). Traditional risk management methodologies focus on the assessment of risks at a singular stage in time and do not address the challenge of managing continuously evolving risk profiles.

Continuous change management in software development has been studied from different perspectives. As an example, Fitzgerald and Stol (2017) proposed a roadmap and agenda for continuous software engineering. In particular, there is a growing interest on security and privacy, and this has motivated work on continuous security (Merkow and Raghavan, 2011), which was proposed to prioritize security throughout the whole software development life cycle.

Previous work discussed risk management in complex and evolving environments. For instance, Omerovic (2016) and Gupta *et al.* (2015) propose managing risk related to accountability, assurance, agility, or financial aspects in multi-cloud applications. Risks analysis is used to guide the selection of cloud service providers. Shrivastava and Rathod (2017) present a risk management framework for distributed agile development, studying risks related to software development life cycle, project management, group awareness, etc. However, they focus on analyzing risk factors that represent a threat to the successful completion of a software development project, rather than risk related to non-functional requirements such as security or privacy. Aslam *et al.* (2017) performed a systematic literature review on risks and control mechanisms for distributed software development. Moran (2014) explicitly tackles issues related to risk management for agile software development. Moran proposes a risk modified kanban board and user story map. Finally, Muntés-Mulero *et al.* (2018) propose a framework to manage risks that supports collaboration, agility, and continuous development. In the MUSA Risk Assessment tool,[3] in order to assess the risks in the different components of an application, the tool asks users to choose among predefined threats that may potentially affect each individual component. Once threats are selected, they are automatically classified in the STRIDE security-oriented framework (*Spoofing identity, Tampering, Repudiation, Information disclosure, Denial of service, and Elevation of privilege*). We take this tool as the baseline for the proposal presented in this chapter. Finally, Khan *et al.* (2017) perform a DFD-based analysis of vulnerabilities and threads, using STRIDE as a framework, in Cyber-physical Systems. Casola *et al.* (2019)

2. RiskWatch: https://www.riskwatch.com. Accessed: 2019-07-18

3. https://musa-project.eu/node/326

propose an almost completely automated process for threat modeling and risk assessment.

None of these contributions tackle the automatic detection of privacy-related vulnerabilities or monitor privacy-related risks for continuous risk management. In general, even for security, risks are not analyzed and monitored continuously, and historical data are not taken into account (Ahmad *et al.*, 2012, Baskerville *et al.*, 2014). Besides, security risks are assessed based on speculation rather than evidence (Shameli-Sendi *et al.*, 2016, Webb *et al.*, 2014). The interaction between analytics capabilities and Information Security Risk Management (ISRM) capabilities can help organizations to perform continuous risk assessments and enable evidence-based decision-making (Naseer *et al.*, 2017).

In parallel, Article 25 in GDPR[4] discusses data protection by design and by default, underlining that considering privacy from the beginning is essential to address privacy successfully. GDPR establishes binding data protection principles, individuals' rights, and legal obligations to ensure the protection of personal data of EU citizens. However, legal measures need to come along with technical measures to protect privacy and personal data in practice. PDP4E H2020 project (Martin and Kung, 2018) focuses on the importance to involve engineers in the loop, for Privacy-by-design (PbD) to be viable.

GeneSIS Modeling Language

GeneSIS (Ferry *et al.*, 2019), developed in the ENACT H2020 project (Ferry *et al.*, 2018), facilitates the development and continuous deployment of TSIS, allowing decentralized processing across heterogeneous IoT, edge, and cloud infrastructures. GeneSIS includes a domain-specific modeling language to model the architecture of TSIS as well as the orchestration and deployment. GeneSIS modeling language is an extension of ThingML (Fleurey and Morin, 2017). We will base our Risk Management methodology in the GeneSIS domain-specific modeling.

LINDDUN

LINDDUN (Wuyts, 2015) is a threat modeling methodology to support risk analysts when addressing privacy risks affecting end users in an application. This methodology provides some guidance to identify and categorize threats under a set of general risks (*Linkability, Identifiability, Non-repudiation, Detectability, Disclosure*

4. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation). Official Journal of the European Union, L119:1–88, May 2016.

of information, Unawareness, and Non-compliance). LINDDUN can be considered the privacy-oriented alternative to the STRIDE framework. In fact, LINDDUN threats are described in the so-called LINDDUN trees which are explicitly connected to STRIDE threats trees (Shostack, 2014). We take this link between privacy and security as the baseline for our proposal in this chapter. The LINDDUN methodology requires to formalize the functionality of the system and their dependencies with respect to personal data [i.e., the Data Flow Diagrams (DeMarco, 1979)]. The notation of a DFD is based upon 4 distinct element types: (i) an external entity (i.e., end users or third-party services that are external to the system), (ii) a data flow (explains data propagation and dependencies between all the functional components), (iii) a data store (i.e., a passive container of information), and (iv) a process (i.e., a computation unit).

8.3 Model-based Continuous Risk Management Methodology

In this section, we propose a methodology for risk management for the creation of TSIS, and we indicate the actors involved in each of those steps. Our methodology (see Figure 8.1) can be summarized in 6 steps:

- **S1: TSIS Assets Definition:** First, we edit or load DFDs representing the functional description of the system. These DFDs are useful to manage risks

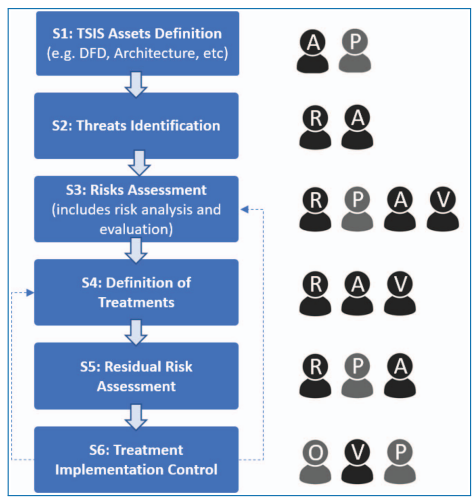

S1: TSIS Assets Definition (e.g. DFD, Architecture, etc)
S2: Threats Identification
S3: Risks Assessment (includes risk analysis and evaluation)
S4: Definition of Treatments
S5: Residual Risk Assessment
S6: Treatment Implementation Control
A P
R A
R P A V
R A V
R P A
O V P

Figure 8.1. Proposed model-based risk management methodology. Roles: (O) Risk Management Owner; (P) Product Owner; (A) Architect; (V) Developer and (R) Risk Analyst.

related to privacy as described by LINDDUN, although they were initially used for security risk control (e.g., in STRIDE). Although DFDs and the elements they consist of represent our primary assets in the risk management process, our proposal entails the use of a second layer of supporting assets in the form of TSIS architecture description. Therefore, in S1, GeneSIS Models (or equivalent architectural models) are also traversed and components are pre-loaded. This step also includes the definition of the vulnerabilities related to assets. Use of the AVD defined later in this section would also be part of this step.

- **S2: Threats Identification:** In this step, users identify threats that may affect the components in the described system.
- **S3: Risk Assessment:** Risk assessment is composed of two different steps: risk analysis, where risks are evaluated in terms of likelihood and consequence, and risk evaluation, where risks are accepted, or they are classified as risks that need to be mitigated.
- **S4: Definition of Treatments:** Mitigation actions are defined in the form of treatments.
- **S5: Residual Risk Assessment:** Once the mitigation controls are defined, the residual risks need to be reassessed. This involves again two steps: risk analysis, where likelihood and consequence are updated after the application of the control(s), and risk re-evaluation, where risks are analyzed again, and they are classified as accepted or further mitigation actions required.
- **S6: Treatment Implementation Control:** Finally, we add a last step in the methodology that goes beyond other previous methodologies. In particular, it involves the monitoring of the effectiveness of the mitigation actions proposed in the previous step. This step requires the connection to data collectors or agents that collect evidences from a monitoring system in order to match them to treatments and risks. Mapping of DFDs and architectural models will be the key for this to be possible.

Note that this methodology is continuous in the sense that new evidence is continuously collected to challenge previous knowledge upon which risk-related decisions were made in the past. While immediate triggering of mitigation actions is not our main focus, conditional options based on the status of the system could be included in S4 to strengthen the continuous approach. Also, the methodology is particularly customized for TSIS, specifically in S1, where GeneSIS models are used as the baseline, and S6, where evidences are collected to consider privacy-related issues. However, the methodology can be easily adapted to other types of systems and used generically.

8.4 Automatic Vulnerability Detection

In order to facilitate an effective identification of privacy-related risks, it is important to make it easy for users to detect those vulnerabilities that expose the system to attacks that may violate data subject's rights. We establish the methodology and bases for the creation of an Automatic Vulnerability Detector (AVD). An AVD uses a set of DFDs to describe an IoT system under development. Based on these DFDs, it is able to detect potential vulnerabilities to kick off the risk analysis process (see step S1 in the previously described methodology).

In order to create the AVD, we use the following methodology: (i) for each DFD component type and for each LINDDUN threat tree, we analyze all the nodes in the tree (containing vulnerabilities) and examine the conditions for those vulnerabilities being relevant in the system; (ii) we create a list of conditions that need to hold for a vulnerability to be effective; (iii) for each instance of each element in every DFD, we collect information about these conditions when defining the system; and (iv) for each component in each DFD related to the system, we filter out vulnerabilities depending on the information collected about these conditions and show those vulnerabilities that are still relevant.

As an example, in Figure 8.2 we show the threat tree related to detectability threats of a data flow in a DFD.[5] A part from the upper node of the tree, which represents the threat under analysis (i.e., Detectability in a data flow), the rest of the nodes refer to vulnerabilities related to this threat. In the figure, we identify several areas from A to E that we will use to define conditions that must hold for the vulnerabilities in those areas to be relevant. Table 8.1 provides an example of some

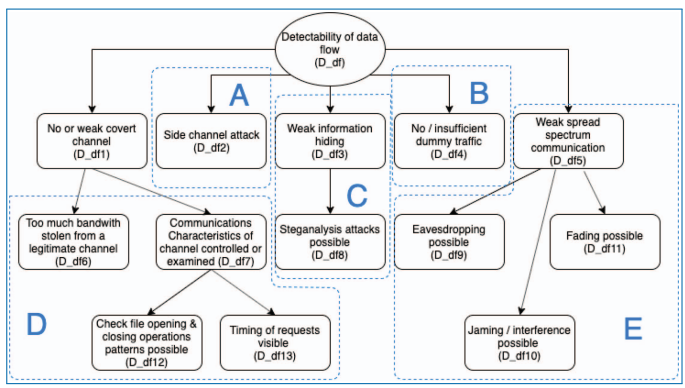

Detectability of data flow (D_df)
A
B
C
D
E
No or weak covert channel (D_df1)
Side channel attack (D_df2)
Weak information hiding (D_df3)
No / insufficient dummy traffic (D_df4)
Weak spread spectrum communication (D_df5)
Too much bandwith stolen from a legitimate channel (D_df6)
Communications Characteristics of channel controlled or examined (D_df7)
Steganalysis attacks possible (D_df8)
Eavesdropping possible (D_df9)
Fading possible (D_df11)
Check file opening & closing operations patterns possible (D_df12)
Timing of requests visible (D_df13)
Jaming / interference possible (D_df10)

Figure 8.2. Analysis of vulnerability scope in the LINDDUN threat tree for detectability in a data flow.

5. LINDDUN Detectability threat tree catalog: www.linddun.org/detectability. Last accessed in Jan 18th 2020.

conditions that must hold for a subset of vulnerabilities extracted from LINDDUN threat trees, related to those areas. For instance, in area C, vulnerabilities related to steganalysis become relevant if the channel is not encrypted. As a second example, in area B, depending on the volume of traffic in the channel, "insufficient dummy traffic" may or may not actually be a vulnerability. In the latter, note also that this condition may change along time, increasing or decreasing the relevance

Table 8.1. Example of conditions for vulnerabilities related to detectability in a data flow (as defined in LINDDUN) to be relevant.

	Vulnerability/ies (Wuyts, 2015)	**Conditions for relevance**	**Rationale**
A	Side channel analysis (D_df2) is based on timing information, power consumption, electromagnetic leaks, etc. It can be used as a source of information which can be exploited to detect the communication.	Do any actions lead to generate footprints in the communication channel? (e.g., Timing information, power consumption, electromagnetic leaks)	If there are no actions generating footprints in the communication channel, the vulnerability is not relevant.
B	Transmitted data can become detectable when there is no or insufficient dummy traffic (D_DF4) sent at some lower layer of the communication network, such that messages fail to appear random for all parties except the sender and the recipient(s).	Is data traffic in the channel very low?	If the traffic is not low, this vulnerability may not be relevant.
C	When weak information hiding techniques (D_df3) are used, steganalysis attacks (D_df8) are possible (detecting messages hidden using steganography).	Channel not encrypted?	If channel is not encrypted, low entropy of unencrypted data facilitates steganography attacks.
D	Detectability of a data flow may happen if the system uses a covert channel in the wrong way (D_df6, D_df7, D_df12, D_df13).	Covert channel used to avoid detectability?	If covert channel is not used these vulnerabilities are not relevant.

(*Continued*)

Table 8.1. Example of conditions for vulnerabilities related to detectability in a data flow (as defined in LINDDUN) to be relevant (continued).

	Vulnerability/ies (Wuyts, 2015)	Conditions for relevance	Rationale
E	The detectability threat can occur because of a weak spread spectrum communication (D_df5), resulting in deficiencies in the establishment of secure communications (allowing eavesdropping (D_df9)), insufficient resistance to natural interference and jamming (D_df10), and insufficient resistance to fading (D_df11).	Is the communication channel wireless?	These vulnerabilities are relevant if the communication is performed on a wireless channel.

of this vulnerability. Therefore, continuous risk management may also include the continuous monitoring of metrics that allow measuring the level of relevance of vulnerabilities or the likelihood of threats to occur.

Note though that Figure 8.2 shows an example for a simple case where the LINDDUN threat tree is not related to any other threat tree. However, in most cases vulnerabilities related to a particular LINDDUN threat trees are related to vulnerabilities detected in other LINDDUN or STRIDE threat trees. Figure 8.3 describes the detail of this connection in the form of a graph, where every node is one of the LINDDUN threat trees (blue nodes) or one of the STRIDE threat trees related to LINDDUN trees (red nodes). As it can be observed, Detectability of a data flow (D_df) is one of the rare cases where the vulnerabilities in the threat tree are not related to those in other trees. In general, most of the trees are related to others. In order to understand the vulnerabilities of a particular component in a DFD, it is important to navigate through these connections. For instance, the analysis of vulnerabilities related to the Identifiability of an entity (I_e) generates a cascade analysis of vulnerabilities that may include I_ds, I_df, ID_df, ID_ds, S_e, and T_p, following directed edges in the graph. Note that edges are colored in gray if threat trees refer to the same DFD element (e.g., $I\_ds \rightarrow ID\_ds$), and they are colored in red if they refer to different types of DFD elements (e.g., $I\_e \rightarrow I\_ds$). For those relationships represented in gray, we assume that we refer to the same element in the same DFDs. For those relationships represented in red, we have explored them one by one and established a rule to propagate the analysis from one

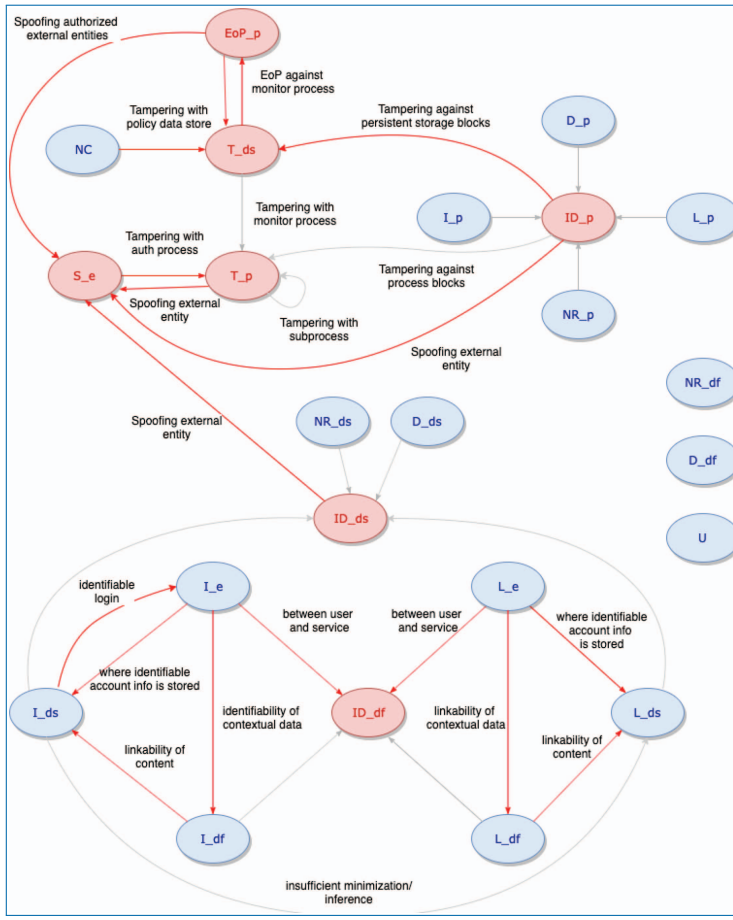


Figure 8.3. Graph describing the relationship between LINDDUN (blue) and STRIDE (red) threat trees. The first part of the ids stand for each of the categories of LINDDUN (Linkability, Identifiability, Non-repudiation, Detectability, Unawareness, and Non-compliance – Disclosure of information is not represented as it is understood as the Information disclosure already captured by STRIDE) and STRIDE (only for those related to LINDDUN: Spoofing identity, Tampering, Information disclosure and Elevation of privilege). The second part after the "_" represents the type of DFD component (entity, data flow, data store, process). Labels in the edges represent indications extracted from LINDDUN descriptions in the corresponding threat trees.

tree to another. For instance, given an entity e in a DFD, for $I\_e \rightarrow I\_ds$, we refer to the data store where the entity credentials of e or other identifiable account information are stored. This means that we will need to ask for the data store names where identifiable account information is stored for each entity. As another example, for $I\_e \rightarrow ID\_df$, we will need to examine all the potential vulnerabilities for all the data flows in the DFD where the origin of the data flow is e. We repeat this analysis for all red arrows in the graph.

8.5 Model-based Risk Management Approach

GDPR establishes a set of duties imposed on the data processors, controllers, and third parties which are aimed at honoring the corresponding data subject's rights. In GDPR, risk is explicitly scoped (Rec. 76) with regard to the *rights and freedoms of the data subject*. In order to understand whether these data subject rights and principles (described in GDPR) are being effectively protected by mitigating existing risks, current approaches tend to analyze the data flow in the system (e.g., LINDDUN and DFDs). However, elements in a DFD tend to be defined at a functional level (e.g., *process to collect profile data from a user* or *process to merge data from two existing data sources*). In this situation, collecting system data to monitor a threat related to the *identifiability of an entity* as defined by LINDDUN, just to take an example, is not obvious, if we do not understand the relationship of a data flow in the application with the architecture of the infrastructure that we monitor. Without a proper connection of these elements to architectural levels that can be monitored, continuous risk management for privacy becomes much more difficult. Most monitoring tools monitor the infrastructure level. These tools may monitor system aspects in a data center or a particular server, tools for network monitoring, etc, and collect metrics.

The main contribution of the approach proposed in this section is the mapping of TSIS architecture models, such as the one proposed by GeneSIS, with data flows in the application under analysis (e.g., DFDs). In S6, we may assume that the elements from both models have been mapped establishing a link between the two types of models. Details on this mapping are provided below. For instance, a NoSQL data store in our system architecture may be mapped to a data store component in a DFD. As another example, a particular process element in a DFD may be mapped to the server the process is running on. As an alternative, it is possible that the mapping is not pre-established before starting the risk analysis process. In this situation, we consider that phase S1 can be extended to allow the user to establish this mapping manually.

Model Mappings

In order to drive the mapping between the architecture model and DFDs, we take as the baseline the connection between privacy-related threats studied in LINDDUN and the vulnerabilities that are related to security aspects of the architecture, as captured by STRIDE. In Table 8.2, we show a sample of vulnerabilities extracted from (Shostack, 2014) related to the STRIDE threats in Figure 8.3. The table shows a brief description and some usual mitigation actions. In the last two columns, we show examples of metrics that could be used for continuous security monitoring

Table 8.2. Sample of the mapping between STRIDE threats (related to LINDDUN threats in Figure 8.2) and related metrics that can be detected from monitoring architectural components. STRIDE tree node extracted from Shostack (2014).

STRIDE Threat	STRIDE tree node	Description	Key mitigation actions	Metrics for continuous monitoring	Monitored components
Spoofing External Entities (*S_e*)	Transit	An attacker copies an authenticator from a non-encrypted channel or tamper with the connection.	Use standard authentication protocols, instead of your own. Use strong encryption and authentication.	Continuous authenticator detector warnings (comparing data in the channel with user database)	Network/ Communication Channels
	Obtain credential from storage at a 3rd party	Reuse of passwords is common. 3rd parties may leak passwords your customers use.	Avoid static passwords. Avoid using e-mail addresses as usernames. Detect brute-force attempts, or increase in successful logins from new locations.	Monitoring of number of brute-force attempts or successful logins from a new location or IP. Continuous user behavior analysis.	Network/ Communication Channels
	Predictable credentials	Predictable usernames or a poor random generator for passwords.	If assigning passwords, use strong randomness.	Detect usernames sent matching actual user names in your internal databases.	Database/Data store

(*Continued*)

Table 8.2. Sample of the mapping between STRIDE threats (related to LINDDUN threats in Figure 8.2) and related metrics that can be detected from monitoring architectural components. STRIDE tree node extracted from Shostack (2014) (continued).

STRIDE Threat	STRIDE tree node	Description	Key mitigation actions	Metrics for continuous monitoring	Monitored components
Information Disclosure at Data Flow (ID_df)	No or Weak Confidentiality (Observing a Message Structure)	Content of a message not protected or weakly protected.	Cryptographic. Use available message protection add-ons or tunneling.	Continuous monitoring message content readability. Can we detect some content structure or detect words with meaning?	Network/ Communication Channels
	No or Weak Confidentiality (Observing a Channel)	No or weak protection of channel contents. Data about the messages can be revealing.	Encrypt the channel. Tunneling.	Continuous channel monitoring content readability. Can we get some structure of the content? Or detect words with meaning?	Network/ Communication Channels
Tampering at Data Store (T_ds)	Bypassing protection rules because of no or weak protection	ACLs, permissions, policies, etc allow people to alter data without a clear justification.	Ensure data is created with appropriate permissions. Change permissions.	Random data editor reporting the number of successful attempts to change data with no or low privileges.	Database/Data store
Information Disclosure of a Process (ID_p)	Timing	Code time execution can reveal confidential information.	Design for cryptography to take constant time.	Monitoring execution time and control variability.	Software components to solve tasks requiring secrecy.

and what type of architectural components are being monitored. We would like to remark that this table is not meant to be an exhaustive list of all the potential threats, but some examples for illustrative purposes that serve the purpose of analyzing which DFD components are to be mapped to which architectural components.

Note that these vulnerabilities are specially relevant in an IoT context. For instance, let us take the example in Table 8.2 related to Information Disclosure of Data Flow. There may be different vulnerabilities associated in particular to TSIS. For instance, if we have a device IoT hub, eavesdropping or interfering the communication between the device and the gateway would be a potential threat. You may also find similar threats in the communication between devices, where data may be read in transit, tampering with the data or overloading the device with new connections or even with the cloud gateways, where eavesdropping or communication interference between devices and gateways may occur.

In general, mappings between architectural models and DFD link:

- Entities in DFDs with the associated network channels in which entity information is transmitted or data stores in which the information is stored;
- Data flows in DFDs with the corresponding network channels;
- Data stores in DFDs with the corresponding databases or data stores used in the system architecture; and
- Processes in DFDs with the corresponding software components executing tasks related to those processes, especially when these tasks entail some level of secrecy.

More formally, we define a set of DFDs, which represent a functional description of the system, $D = \{D_1, D_2, \ldots, D_n\}$, as a set of graphs $D_i = (DV_i, DE_i)$, where $DV_i = \{e_1, \ldots, e_m\}$ are the elements in the DFD representing elements of the system architecture at the functional level. Each element e_i can be an *external entity*, *data store*, or a *process*. $DE_i = \{f_1, \ldots, f_k\}$ represent the *data flows* connecting elements in DV_i. We define a graph $A = \{V_A, E_A\}$ where $V_A = \{c_1, \ldots, c_m\}$ represent the architectural components at the technical level represented in a model such as those created by GeneSIS and $E_A = \{l_1, \ldots, l_k\}$ represent the connections between any two components (c_i, c_j) in the system architecture. Note that components can be both software or hardware components. We define a mapping between the two models D_i and A as a function $\mathcal{M} : DV_i \cup DE_i \rightarrow V_A \cup E_A$.

Exploiting Model Mappings

Once the mapping is established, then a final step is necessary to link metrics, such as those presented as examples in Table 8.2, to the related risks or mitigation

actions defined in S3 and S4. Once the risk model is developed for each DFD, we propose two ways to exploit the established mappings to enable continuous risk management:

- *Automatic likelihood recalculation*: During risk assessment in S3, an initial likelihood and consequence values are determined for each risk. However, a basic principle for continuous risk management is that conditions in our system may change as time goes by. Let e be an element in a DFD D_i, $e \in DV_i \cup DE_i$, considered an asset in the risk management process. In S3, we define risks $r_1, \ldots, r_j$ associated to e. Given a system architecture defined through a model $A = \{V_A, E_A\}$, we define a set of metrics $m_1^c, \ldots, m_h^c$ for each component $c \in V_A \cup E_A$. Automatic likelihood recalculation involves defining a function f for each risk r related to each asset e such that $c = \mathcal{M}(e)$ and $f(m_1^c, \ldots, m_h^c)$ provide a new value for the likelihood or risk r. With this, we connect the metrics defined for the architectural components and use them to calculate the likelihood of risks connected to elements of the functional description of a system, in the corresponding DFD.
- *Treatment effectiveness control*: A parallel approach to enable continuous risk management, involves the definition of a function g for each r related to an asset $e \in DV_i \cup DE_i$ such that $c = \mathcal{M}(e)$ and $g(m_1^c, \ldots, m_h^c)$ represents a KPI that needs to be satisfied for a risk to be considered effectively mitigated. If g exceeds a particular threshold, then a warning needs to be issued for the risk plan including mitigation actions to be reviewed.

8.6 Conclusions

Handling privacy-related risks is not well understood in the IoT context. There is a lack of mechanism to effectively detect vulnerabilities related to privacy and to collect meaningful evidences and continuously monitor risks. We need to find the intersection between data protection challenges and the enabling of TSIS. We have presented a first step towards the enactment of continuous privacy-related risk control for TSIS, leveraging the link offered by LINDDUN between privacy and security threat categories and combining functional and architectural models. As future work, we need to establish a stronger connection between privacy threat models like LINDDUN and the actual requirements imposed by GDPR, as current threat models were devised before GDPR and they may not fully cover all derived requirements. In particular, the relationship between data subject rights and GDPR

principles and LINDDUN categories is unclear. Besides, we need to understand how other privacy-related evidences can be collected beyond those collected for security purposes. In particular in TSIS, where complex and distributed systems increase system weaknesses and the attack surface.

Acknowledgments

This work is supported by the European Commission through the *ENACT* project under Project ID: 780351 and the *PDP4E* project under Project ID: 787034. David Sanchez-Charles was part of the staff of Trialog SA by the time this article was written. We would like to thank A. Kung and the rest of Trialog's team for their continuous support in this research.

References

Ahmad, A., J. Hadgkiss, and A. B. Ruighaver. 2012. "Incident response teams–Challenges in supporting the organisational security function." *Computers & Security*. 31(5): 643–652.

Alberts, C. J. and A. Dorofee. 2002. *Managing Information Security Risks: The Octave Approach*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc. ISBN: 0321118863.

Aslam, A., N. Ahmad, T. Saba, A. S. Almazyad, A. Rehman, A. Anjum, and A. Khan. 2017. "Decision Support System for Risk Assessment and Management Strategies in Distributed Software Development." *IEEE Access*. 5: 20349–20373. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2017.2757605.

Baskerville, R., P. Spagnoletti, and J. Kim. 2014. "Incident-centered information security: Managing a strategic balance between prevention and response." *Information & Management*. 51(1): 138–151.

Boehm, B. and R. Turner. 2003. *Balancing agility and discipline: A guide for the perplexed, portable documents*. Addison-Wesley Professional.

Brooks, S., S. Brooks, M. Garcia, N. Lefkovitz, S. Lightman, and E. Nadeau. 2017. *An introduction to privacy engineering and risk management in federal systems*. US Department of Commerce, National Institute of Standards and Technology.

Casola, V., A. De Benedictis, M. Rak, and U. Villano. 2019. "Toward the automation of threat modeling and risk assessment in IoT systems." *Internet of Things* 7: 100056.

DeMarco, T. 1979. "Structure analysis and system specification." In: *Pioneers and Their Contributions to Software Engineering*. Springer, pp. 255–288.

Ferry, N., P. H. Nguyen, H. Song, P.-E. Novac, S. Lavirotte, J.-Y. Tigli, and A. Solberg. 2019. "GeneSIS: Continuous Orchestration and Deployment of Smart IoT Systems." In: the proceedings of the IEEE COMPSAC conference, Milwaukee, USA, July 15–19. Springer.

Ferry, N., A. Solberg, H. Song, S. Lavirotte, J.-Y. Tigli, T. Winter, V. Muntés-Mulero, A. Metzger, E. R. Velasco, and A. C. Aguirre. 2018), 'ENACT: Development, Operation, and Quality Assurance of Trustworthy Smart IoT Systems." In: *International Workshop on Software Engineering Aspects of Continuous Development and New Paradigms of Software Production and Deployment*. pp. 112–127.

Fitzgerald, B. and K. Stol. 2017. "Continuous software engineering: A roadmap and agenda." *Journal of Systems and Software*. 123: 176–189. DOI: 10.1016/j.jss.2015.06.063. URL: http://dx.doi.org/10.1016/j.jss.2015.06.063.

Fleurey, F. and B. Morin. 2017. "ThingML: A generative approach to engineer heterogeneous and distributed systems." In: *2017 IEEE International Conference on Software Architecture Workshops (ICSAW)*. pp. 185–188.

Gupta, S., V. Muntés-Mulero, P. Matthews, J. Dominiak, A. Omerovic, J. Aranda, and S. Seycek. 2015. "Risk-driven framework for decision support in cloud service selection." In: *2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*. pp. 545–554.

Karabacak, B. and I. Sogukpinar. 2005. "ISRAM: Information Security Risk Analysis Method." *Comput. Secur.* 24(2), 147–159. issn: 0167-4048. DOI: 10.1016/j.cose.2004.07.004. URL: http://dx.doi.org/10.1016/j.cose.2004.07.004.

Khan, R., K. McLaughlin, D. Laverty, and S. Sezer. 2017. "STRIDE-based threat modeling for cyber-physical systems." In: *2017 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*. 1–6. DOI: 10.1109/ISGTEurope.2017.8260283.

Lund, M. S., B. Solhaug, and K. Stølen. 2010. *Model-driven risk analysis: the CORAS approach*. Springer Science & Business Media.

Martin, Y.-S. and A. Kung. 2018. "Methods and tools for GDPR compliance through privacy and data protection engineering." In: *2018 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. 108–111.

Merkow, M. and L. Raghavan. 2011. "An Ecosystem for Continuously Secure Application Software." RUGGED Software, CrossTalk March/April.

Moran, A.. 2014. *Agile Risk Management*. Springer International Publishing. isbn: 978-3-319-05007-2.

Muntés-Mulero, V., O. Ripolles, S. Gupta, J. Dominiak, E. Willeke, P. Matthews, and B. Somosköi. 2018. "Agile risk management for multi-cloud software development." *IET Software*. 13(3): 172–181.

Naseer, H., G. Shanks, A. Ahmad, and S. Maynard. 2017. "Towards an Analytics-Driven Information Security Risk Management: A Contingent Resource Based Perspective." *Procs of ECIS* 2017: 2645–2655.

Omerovic, A. 2016. "Supporting Cloud Service Selection with a Risk-Driven Cost-Benefit Analysis." In: *Advances in Service-Oriented and Cloud Computing*. Ed. by A. Celesti and P. Leitner. Cham: Springer International Publishing. 166–174. isbn: 978-3-319-33313-7.

Shameli-Sendi, A., R. Aghababaei-Barzegar, and M. Cheriet. 2016. "Taxonomy of information security risk assessment (ISRA)." *Computers & Security*. 57: 14–30.

Shostack, A.. 2014. *Threat modeling: Designing for security*. John Wiley & Sons.

Shrivastava, S. V. and U. Rathod. 2017. "A risk management framework for distributed agile projects." *Information and Software Technology*. 85: 1–15. issn: 0950-5849. DOI: https://doi.org/10.1016/j.infsof. 2016.12.005. URL: http://www.sciencedirect.com/science/article/pii/S0950584916304815.

Webb, J., A. Ahmad, S. B. Maynard, and G. Shanks. 2014. "A situation awareness model for information security risk management." *Computers & Security*. 44: 1–15.

Wuyts, K. 2015. "Privacy Threats in Software Architectures."

Yan, Z., P. Zhang, and A. V. Vasilakos. 2014. "A survey on trust management for Internet of Things." *Journal of Network and Computer Applications*. 42: 120–134.

DOI: 10.1561/9781680836837.ch9

Chapter 9

Data Protection Compliance Requirements for the Internet of Things

By Luca Bolognini, Sébastien Ziegler, Pasquale Annicchino, Francesco Capparelli and Alice Audino

9.1 Introduction

With the progressive implementation of new technologies, more and more interconnected, we are surrounded by an increasingly synchronized, delocalized, and correlated environment, animated by a continuous flow of data generated by our choices and behaviors.

The disruptive advent of IoT technologies has profoundly changed the relationship between human beings and "things." It is difficult to make a distinction between online and offline dimensions, because most of the devices used in everyday life are connected and interact with the surrounding reality in a continuous exchange of information between material and virtual environments.[1]

It can be observed, therefore, that a new challenge has opened up for the protection of the rights and freedoms of Data Subjects and stakeholders in the new technological environment that has changed since the broader adoption of IoT.[2]

1. L. Floridi, *The Ethics of information*, Oxford University Press, 2015.

2. EUROPEAN COMMISSION, The Internet of Things Opportunities and Challenges, 2015. Document available at the following link: https://www.europarl.europa.eu/RegData/etudes/BRIE/2015/557012/EPRS_BRI(2015)557012_EN.pdf. The document defines IoT as "a global, distributed network

Existing digital and electronic devices have implied the need to strongly focus on personal data protection safeguards. The more the technologies have evolved, the more it has been necessary to establish a legislative framework to protect a "personal" and "private" space. These needs have been at the heart of debates in the Parliament and in the Commission of the European Union, bringing to the adoption of a legal instrument to be valid in all the European Union Member States, the Regulation n. 679/2016, the General Data Protection Regulation (GDPR). This Regulation was introduced into an existing International and European normative framework protecting fundamental rights and freedoms, which includes, among others, the Universal Declaration of Human Rights and the Charter of Fundamental Rights of the European Union.

Even if the Regulation does not explicitly refer to IoT Devices or the IoT System, the principles emerging from the GDPR—first of all, the ones of purpose limitation, storage limitation, lawfulness, transparency, fairness, integrity and confidentiality, and data protection-by design/by default—are adaptable to any technological development and deployment, IoT technologies included.

The present document proposes some brief considerations, without claiming to be exhaustive, on the issues of Data Subject awareness and accountability in the IoT field, with a focus on technology developers in the light of the GDPR and cybersecurity international best practices.

9.2 IoT and General Data Protection Regulation: Awareness as a Key Safeguarding Factor

9.2.1 Awareness and Data Protection

Data subjects' awareness is a prerequisite in order to enhance their confidence in new technologies, as well as to ensure that the individuals can always choose, freely and autonomously, whether allowing or not the collection and further processing of their data in an IoT environment.

Data protection, privacy, and security are the main key factors to increase trust. The challenge is, then, to develop technologies that are, by design, inherently privacy-preserving and transparent, in order to empower end users (and more in general, the end target of individuals) to understand and to be informed of (and, where appropriate, to control over) the use of their personal data. The interaction

(or networks) of physical objects that are capable of sensing or acting on their environment, and able to communicate with each other, other machines or computers". See also ENISA, Baseline Security Recommendations for IoT in the context of Critical Information Infrastructures, 2017. The document refers to the concept introduced by *Kevin Ashton*: "a wide ecosystem where interconnected devices and services collect, exchange and process data in order to adapt dynamically to a context."

between individuals and IoT Devices or the IoT System has been studied in order to better understand practical and actual dynamics of relationship between individuals and objects; as a result, it emerged the need to ensure a human-centric IoT environment, as the EU project GHOST[3] demonstrates.

A new IoT project may involve users and Data Subjects as actors or as factors. Most of the IoT solutions are considering the possible *factorization* of people: people as an entity which interacts with other sensors. *Factorization* could bring to limit Data Subjects' rights and freedom, with the risk of *de facto* downgrading their role to the same level of "objects." A way to empower individuals, without *factorizing* them, is to educate and train them on the IoT ecosystem features, risks, and safeguards, so as to promote the creation and perception of an environment that they can consider "safe" and "manageable."

A better education would also imply an increasing awareness of Data Subjects' skills and rights, related to the adoption of IoT tools, as well as a major attention by developers and producers to IT security and other technological features, in order to protect, by default, individuals from possible violations and to enable them to receive adequate information and to exercise their rights.

Moreover, the way in which personal information is collected and managed, in an IoT scenario, may affect not only end users' rights but also the rights of third subjects which are observed and recorded by the devices.

The processing of personal data carried out through IoT systems is certainly included in the material scope of application of the GDPR, which sets forth some conditions for the lawfulness of the processing as listed in Article 6:

"[…] *processing shall be lawful only if and to the extent that at least one of the following applies:*

(*a*) *The Data Subject has given consent to the processing of his or her personal data for one or more specific purposes;*
(*b*) *Processing is necessary for the performance of a contract to which the Data Subject is party or in order to take steps at the request of the Data Subject prior to entering into a contract;*
(*c*) *Processing is necessary for compliance with a legal obligation to which the Controller is subject;*
(*d*) *Processing is necessary in order to protect the vital interests of the Data Subject or of another natural person;*
(*e*) *Processing is necessary for the performance of a task carried out in the public interest or in the exercise of official authority vested in the Controller;*

3. GHOST Safe-Guarding Home IoT Environments with Personalised Real-time Risk Control: "D3.9: Trials use case specification and report" (1st release).

(*f*) *Processing is necessary for the purposes of the legitimate interests pursued by the Controller or by a third party, except where such interests are overridden by the interests or fundamental rights and freedoms of the Data Subject which require protection of personal data, in particular where the Data Subject is a child.*"

There are six available legal bases for the processing. No single basis is necessarily more suitable than the others, due to the fact that the most appropriate basis will depend on the purpose of the data processing and on the possible relationship between Controllers and Data Subjects.

Among the various bases stated in the aforementioned Art. 6 of the GDPR, the one that necessarily implies a good degree of awareness and information of the Data Subjects is described in letter (a): consent.

In particular, according to Article 7 of the GDPR:

1. "*Where processing is based on consent, the Controller shall be able to demonstrate that the Data Subject has consented to processing of his or her personal data.*
2. *If the Data Subject's consent is given in the context of a written declaration which also concerns other matters, the request for consent shall be presented in a manner which is clearly distinguishable from the other matters, in an intelligible and easily accessible form, using clear and plain language. Any part of such a declaration which constitutes an infringement of this Regulation shall not be binding.*
3. *The Data Subject shall have the right to withdraw his or her consent at any time. The withdrawal of consent shall not affect the lawfulness of processing based on consent before its withdrawal. Prior to giving consent, the Data Subject shall be informed thereof. It shall be as easy to withdraw as to give consent.*
4. *When assessing whether consent is freely given, utmost account shall be taken of whether, inter alia, the performance of a contract, including the provision of a service, is conditional on consent to the processing of personal data that is not necessary for the performance of that contract.*"

Article 4 paragraph 1 n. 11 GDPR rules that "*consent of the Data Subject means any freely given, specific, informed and unambiguous indication of the Data Subject.*" However, as provided by the WP29 in the opinion adopted in 2014 on the "*Recent Developments on the Internet of Things*":[4] "*In many cases, the user may not be aware*

4. ARTICLE 29 DATA PROTECTION WORKING PARTY This Working Party was set up under Article 29 of Directive 95/46/EC. It is an independent European advisory body on data protection and privacy. Its tasks are described in Article 30 of Directive 95/46/EC and Article 15 of Directive 2002/58/EC. The secretariat is provided by Directorate C (Fundamental Rights and Union Citizenship) of the European Commission, Directorate General Justice, B-1049 Brussels, Belgium, Office No MO-59 02/013. Website: http://ec.europa.eu/justice/data-protection/index_en.htm14/ENWP223 Opinion 8/2014 on the Recent Developments on the Internet of Things Adopted on 16 September 2014.

of the data processing carried out by specific objects. Such lack of information constitutes a significant barrier to demonstrating valid consent under EU law, as the Data Subject must be informed" pursuant to Article 13 and Article 14 of the GDPR.

In all cases of different legal bases of personal data processing, other than Data Subjects' consent (think about the data processing in the legitimate interest of the Controller or in the public interest), the challenge would be that of adequately informing Data Subjects and to make them as aware as possible of the activities carried out and of the related risks to their rights and freedoms.

9.2.2 Awareness of Data Subject as an Instrument for Opt-in and Free Choices

The processing of information by IoT technologies can involve natural persons. In this sense, Articles 13 and 14 of the GDPR provide the obligation to inform Data Subjects about the personal data lifecycle, while Recital 58 of the GDPR underlines that the principle of transparency requires that "*any information addressed to the public or to the Data Subject be concise, easily accessible and easy to understand, and that clear and plain language and, additionally, where appropriate, visualisation be used*".

In addition, Recital 60 underlines that the obligation to inform comes from the application of the principles of fairness and transparency of processing. Precisely, the latter principle is also the logical basis for a conscious Data Subject's choice.

The "*Guidelines on transparency under Regulation 2016/679*"[5] issued by WP29 clarified that this principle applies in three main cases (par. 7):

(1) In the information provided to the Data Subjects;
(2) The way in which Data Controllers communicate with Data Subjects the information;
(3) The way in which Data Controllers allow the exercise of rights by the Data Subjects.

Therefore, the *ratio legis* of this principle is to allow Data Subjects to have control over their data and thus to make a choice about it.

In particular, paragraph 10 of the document states that: "*the Data Subject should be able to determine in advance what the scope and consequences of the processing entails*

5. ARTICLE 29 DATA PROTECTION WORKING PARTY This Working Party was set up under Article 29 of Directive 95/46/EC. It is an independent European advisory body on data protection and privacy. Its tasks are described in Article 30 of Directive 95/46/EC and Article 15 of Directive 2002/58/EC. The secretariat is provided by Directorate C (Fundamental Rights and Union Citizenship) of the European Commission, Directorate General Justice, B-1049 Brussels, Belgium, Office No MO-59 02/013. Website: http://ec.europa.eu/newsroom/Article29/news.cfm?itemtype=1358&tpaid=6936 17/EN WP260 rev.01 Article 29 Working Party Guidelines on transparency under Regulation 2016/679 Adopted on 29 November 2017. As last Revised and Adopted on 11 April 2018.

and that they should not be taken by surprise at a later point about the ways in which their personal data has been used."

The Data Controller would therefore be obliged to:

- Provide individuals with information including the purposes for the processing of their personal data, the retention periods for personal data processed, and with whom personal data will be shared;
- Provide privacy information to individuals at the time the IoT Data Controller collects their personal data;
- Obtain personal data from other sources; the IoT Data Controller shall provide individuals with privacy information within a reasonable period and no later than one month; there are few circumstances when the IoT Data Controller does not need to provide people with privacy information, such as, if an individual already has the information or if it would involve a disproportionate effort to provide information to them;
- Provide information to people in a concise, transparent, intelligible, easily accessible manner, and it shall use clear and plain language;
- Regularly review, and where necessary, update its privacy information. The IoT Data Controller shall provide the individuals with any information related to the new uses of their personal data, before the IoT Data Controller starts the processing.

However, in this context, where information is exchanged between Data Subjects and "Things" and between "Things" and "Things," it is in many cases very difficult to make Data Subjects aware of the life cycle of their data. For this reason, it is not always that easy and effective to provide an *ex ante* information notice to Data Subjects, in an IoT scenario, while it could result more effective and efficient to provide relevant information, in a simplified way, after data collection or, anyway, in a non-traditional modality.

A way of considering the matter would be, then, the possibility of publishing a concise, schematic information, similar to labels on the packaging of medicines or food, to be affixed in proximity of the inter-connected arrangement. This signal or label […] like the one used for snacks […] would specify which data are being and/or have been processed and by whom and would show to the Data Subjects how they could exercise their data rights.

The solution here suggested could be particularly advantageous because the information on the label/signal will also help non-expert people to understand which categories of information are involved in the purposes and in the logic that rules the processing, by clarifying the structure (data, sources, criteria) of the processing itself, so as to better implement the principles of lawfulness, fairness, and transparency (pursuant to Article 5 paragraph 1, letter (a) of the GDPR). IoT

could be equipped with well-visible interfaces, displaying such labels. For example, an image sensor and an IoT object could be equipped with a screen, to increase awareness on the presence of the sensor and increase user confidence in using objects that gather data.

Another option, connected with the information just indicated above, could result in the obligation to alert individuals about the presence of IoT sensors in a given environment. Specifically, following the provision set forth by Article 12 paragraph 8 of the GPDR, where it is expected that "*The Commission shall be empowered to adopt delegated acts in accordance with Article 92 for the purpose of determining the information to be presented by the icons and the procedures for providing standardized icons*," Solutions which provide for the use of icons or figurative representations of other kind, easily understandable by the interested parties, may be considered.

This "alert signals" solution could be even more helpful – in all cases in which Data Subjects are not "end users," but simply "non-users," since there are no interfaces and no features for interactivity between things and persons – in order to prevent negative effects caused by IoT deployments.

Finally, a third option consists in using a dedicated smart phone application to inform the data subjects about the presence of IoT in their vicinity. Such approach has been researched and implemented in the European Large Scale Pilot on IoT for smart cities, the H2020 Synchronicity European research project.[6] A dedicated application, titled Privacy App (www.privacyapp.info), has been developed.

The Privacy Application enables users to discover nearby IoT Devices or the IoT System on an interactive map. For each IoT device, the app provides the available information related to the purpose of the data processing, the controllers and processors involved, as well as information on retention period and cross-border transfer. The user can also add new IoT Devices or the IoT System on the map and request for more information. More importantly, the user can directly contact the Data Protection Officer of the data controller of the IoT device.

9.2.3 IoT Features Enabling Opting-out and Exercise of Data Subjects' Rights

Awareness is a prerequisite of Data Subjects' freedom of choice, as underlined above; a robust level of awareness may enable individuals to prevent their data collection (*ex ante*). However, awareness can empower individuals to control over their data after processing and to exercise their rights in an opt-out approach (*ex post*).

6. The Privacy App was developed by Mandat International in the context of Synchronicity, the H2020 European Large Scale Pilot on the Internet of Things for Smart Cities: https://synchronicity-iot.eu/

Hereinafter, we summarize the provisions of Articles 15 to 22 of the GDPR, which contain the rights of the Data Subject (all terms in capital letters refer to the definitions given in the GDPR):

- Right of access: according to the right of access pursuant to Article 15 of the GDPR, the Data Subject has the right to obtain the confirmation as to whether Personal Data concerning him or her is being Processed. The right of access can imply, if requested by the Data Subject, the right to receive a copy of the personal data being processed.
- Right to erasure is defined as a right to ask for the deletion of personal data. According to the right to erasure under Article 17 of the GDPR, the Data Subject has the right to request for and obtain the erasure of Personal Data or the anonymization of such data, provided that it takes place by means of techniques which avoid the re-identification of the Data Subject. Subject to the assessment of the conditions set forth in Article 17 paragraph 1 of the GDPR, the Data Controller, having considered the technologies at its disposal and the costs, has to promptly notify the erasure, unless it involves a disproportionate effort, to all Recipients to whom Personal Data have been transferred. The Data Controller communicates to the Data Subject the erasure once completed.
- The right to restrict processing requires that personal data are "marked," without being processed for purposes other than mere storage, pending further determinations; therefore, it is advisable for the Data Controllers to include in their information systems (electronic or otherwise) suitable measures for this purpose. The request for restriction of the Processing pursuant to Article 18 of the GDPR, with the exception of storage, implies the prohibition of any type of Processing of the Data Subject's Personal Data unless the following circumstances apply: the Data Subject's Consent has been given; the Processing is necessary for the establishment, exercise, or defense of legal claims; the Processing is necessary to protect any other natural or legal person's rights; a substantial public interest applies. The Data Controller must promptly notify the request for restriction pursuant to Article 18 of the GDPR to any other Recipient to whom Personal Data were communicated unless it involves a disproportionate effort.
- The right to rectification: the interested party has the right to obtain from the Data Controller the rectification of inaccurate personal data concerning him/her without unjustified delay. The Data Subject has the right to obtain rectification/correction of his/her inaccurate Personal Data under Article 16 of the GDPR. The Data Controller, if it is possible and unless it involves a disproportionate effort, has to notify the rectification/integration to each Recipient to which it has communicated the Personal Data; once the Data

Processor rectifies or integrates Personal Data, it is necessary to notify the Data Subject promptly.

- Right to portability: the Data Subject shall have the right to receive the personal data concerning him or her, which he or she has provided to a controller, in a structured, commonly used, and machine-readable format and have the right to transmit those data to another controller without hindrance from the controller to which the personal data have been provided, where:

  (a) the processing is based on consent or on a contract; and
  (b) the processing is carried out by automated means.

- Right to object: the Data Subject shall have the right to object, at the time of the processing of personal data, to what is based on the point (e) or (f) of Article 6 paragraph 1 of the GDPR, including Profiling based on those provisions. The Data Subject may request to object to Process his/her Personal Data, including Profiling (except for) in the following cases: the Processing is necessary to comply with a relevant public interest or in the exercise of official authority vested in the Data Controller; the Processing is necessary for the purposes of the legitimate interests pursued by the Controller or by a third party, except where such interests are overridden by the interests or fundamental rights and freedoms of the Data Subject which require protection of personal data, in particular where the Data Subject is a child. In such cases, the request for objection must be motivated and may be refused by the Data Controller in the following cases: the Data Controller demonstrates compelling legitimate reasons for Processing that override Data Subject's interests, rights, and freedoms; Processing is necessary for the establishment, exercise, or defense of legal claims. Save the above cases, the Data Controller refrains from further Processing Personal Data. The request for objection determines the termination of the Processing. Personal Data are deleted and anonymized pursuant to Article 17 paragraph 1 of the GDPR, unless one of the exceptions above applies.
- Automated individual decision-making, including Profiling: each individual has the right not to be subject to decisions based solely on automated Processing, including Profiling, which produce legal effects concerning him or her or similarly significantly affect him or her. Furthermore, the Data Controller has to guarantee to the Data Subject a human intervention and/or to express his or her point of view, and as well as to contest the decision. In case an Automated individual decision-making is carried out, the Controller has the obligation to provide the Data Subject with meaningful information about the logic involved, as well as the relevance and the envisaged consequences of such Processing for him/her.

9.3 The Principles of Accountability, Data Protection by Design and by Default as Indirect Requirements for IoT Technology Producers and Controllers

9.3.1 Controllers vs Producers in IoT: Direct and Indirect GDPR Requirements

Articles 5 and 24 of the GDPR set one of the fundamental and innovative principles of the Regulation, the essence of the legislator's change of approach with the GDPR, establishing the principle of accountability of the Data Controller. It is the Controller's task + the action to "*implement appropriate technical and organisational measures to ensure, and be able to show, that the processing is carried out in accordance with the Regulation.*" Furthermore, Article 24 paragraph 2 of the GDPR provides that "*these measures shall be reviewed and updated as necessary,*" which is intended to reaffirm the central role of the Controller and its responsibility in adopting technical and organizational measures.

The principle of accountability is key, in order to better understand the nature of the activities that the Data Controller must carry out from the earliest stages of design of new processes, products or services.

Whoever undertakes to process, must ensure that the data must be "*accountable.*" This means, on the one hand, that Controllers will be responsible for the identification and choice of the legal bases, the procedures, the modalities of implementation, the security measures and the operations, which involve the processing of the data, but also, on the other hand, that Controllers must be able to demonstrate that their actions comply with the provisions of the GDPR, especially with regard to the "*effectiveness of the measures,*" keeping records and documentation of such assessments and choices.

Measures to be put in place by Controllers and Processors shall be "*adequate*," but Article 24 of the GDPR does not specify what measures can be adequate: it is up to Controllers the assessment of possible fitting technical and security measures, on a case-by-case basis.

The "*one-size-fits-all checklist*" approach is therefore replaced by a by-design approach, permeated with contextualization and risk-specific analysis.

The European legislator's new approach is typical of international standards on the management of information systems. The determinations deriving from the knowledge of the internal dynamics of an organization, and its processes/activities, are potentially more effective than the choices made for the mere purpose of complying with a norm.[7]

7. The approach of the International Standard Organization already in 2013, with the ISO/IEC 27001 Standard, focuses on risk assessment as a source from which to derive the implementation of technical and organizational security measures.

The application of the principle of accountability, therefore, should bring IoT designers, producers, and deployers to weigh and objectively justify their choices, in matter of personal data processing, while at the same time, allowing them to devote the necessary resources and effort to achieve the highest level of security and safety for individuals, considering their dynamics and processes, and not to only refer to a list of abstract requirements.

With regard to effectiveness, WP29 in op. 3/10[8] highlights that "*there are various methods available to Data Controllers to assess the effectiveness (or ineffectiveness) of the measures. For the processing of larger, more complex and higher risk data, internal and external audits are common verification methods. The way audits are conducted can also vary, from full audits to negative audits (which can take different forms)*," so to evaluate the effectiveness of the measures. Depending on the type of data processing activities carried out, it will be needed to use both internal and external means of evaluation. Article 24 paragraph 1 of the GDPR, additionally, expressly provides that the measures adopted shall be periodically "*reviewed and updated where necessary.*" This is particularly relevant for IoT, in light of the great variety of possible deployment scenarios of IoT solutions and of the risks to rights and freedoms of individual that may derive from them.

This approach is, furthermore, compatible with the ISO/IEC 27001 standard, which obliges those who provide certification of conformity to the standard to carry out repeated risk assessments over time, planned in such a way as to discover and adequately mitigate previously ignored risks and to further mitigate the risks already treated.

The principle of accountability is directly linked to the principles set out in Article 25 of the GDPR, the "data protection by default principle" and the "data protection by design principle."

The latter opens to a pragmatic view: taking into account the state of the art, the cost of implementation and the nature, scope, context and purposes of processing as well as the risks of varying likelihood and severity for rights and freedoms of natural persons posed by the processing, the Controller shall, both at the time of the determination of the means for processing and at the time of the processing itself, implement appropriate technical and organizational measures, such as pseudonymization, which are designed to implement data-protection principles, or data minimization, in an effective manner and to integrate the necessary safeguards into the processing in order to meet the requirements of the GDPR and protect the rights of Data Subjects.

8. ARTICLE 29 DATA PROTECTION WORKING PARTY 17/EN WP260 Guidelines on transparency under Regulation 2016/679.

The "data protection by default principle" finds substance in all those measures, technical and organizational, functional to ensure that only personal data necessary for the purposes pursued are processed. That obligation applies to the amount of personal data collected, the extent of their processing, the period of their storage and their accessibility. In particular, such measures shall ensure that by default personal data are not made accessible without the individual's intervention to an indefinite number of natural persons.

In the light of the above considerations, in conclusion, it should be noted that the GDPR, in its Articles 5-24-25 as analyzed above, does not provide for a merely formal fulfillment of data protection obligations; quite the opposite, the Regulation calls for a real assessment, concrete and substantial, by those who are responsible for the processing of personal data. In this regard, the recent EDPB guidelines on the topic have specified that (par. 8): "The term *measures* can be understood in a broad sense as any method or means that a Controller may employ in the processing. These measures must be appropriate, meaning that they must be suited to achieve the intended purpose, i.e. they must be fit to implement the data protection principles effectively by reducing the risks of infringing the rights and freedoms of Data Subjects. The requirement to *appropriateness* is thus closely related to the requirement of *effectiveness*".[9]

The GDPR introduces, of course, a cultural change that will inevitably impact also IoT developers and manufacturers, even if it has to be taken into account that the "accountability principle," the "data protection by default principle" and the "data protection by design principle" refer to actual Data Controllers, while, according to Recital 78 of the GDPR, "*producers of the products, services and applications* ***should be encouraged*** [but not obliged, authors' note] *to take into account the right to data protection when developing and designing such products, services and applications and, with due regard to the state of the art, to make sure that Controllers and processors are able to fulfil their data protection obligations.*"

Notwithstanding this, the GDPR combines the mild "encouragement" of producers with the principle of accountability to be respected by Data Controllers, thus elevating data protection by design from an option to a strict parameter of compliance to be assessed by Controllers in the selection of products and/or services providers. And the mentioned Recital 78 of the GDPR provides that "*the principles of data protection by design and by default should also be taken into consideration in the context of public tenders.*"

9. EDPB, Guidelines 4/2019 on Article 25 Data Protection by Design and by Default. In particular, the concept of effectiveness is extremely connected with the accountability principle of Article 5 paragraph 2 of the GDPR. Indeed, the Data Controllers must be able to demonstrate that they have implemented dedicated measures to protect these principles, and that they have integrated specific safeguards that are necessary to secure the rights and freedoms of Data Subjects.

In conclusion, the compliance with the GDPR principles can be considered as an "indirect requirement" for IoT technology producers, even if they do not directly process personal data on behalf of Controllers.

To this end, new certification mechanisms, guidelines and standards will be welcome, in order to direct the application of such principles in the field of IoT, both from the side of producers and of Data Controllers and Processors.

9.3.2 Cybersecurity Measures for IoT: Recommendations

IoT Devices or the IoT System can simplify our lives and their adoption is constantly increasing; unfortunately, such devices and applications are not always safe from the cybersecurity point of view.

The growing number of devices which are connected to the network also raises the amount of possible vulnerabilities and access points for potential IT attacks. The fact of being networked objects makes IoT Devices or the IoT System vulnerable to cyber threats.

In this sense, in 2019 the European Telecommunications Standards Institutes (hereinafter referred to as "ETSI") issued a document "to support all parties involved in the development and manufacturing of consumer IoT with guidance on securing their products."[10] In particular, the document proposed solutions consistent with the GDPR approach.

From the abovementioned source, it is possible to extract several recommendations of security measures to enhance security in IoT systems and, thus, Data Subjects' engagement and trust. In the Table 9.1 below, we can list the cybersecurity measures envisaged by the ETSI Document.

Table 9.1. Cybersecurity measures for an IoT device and an IoT system (Source: ETSI).

The IoT Devices or the IoT System requires at least one administrative user, that is, a user having the ability to operate with elevated privileges inside the IoT Devices or the IoT System (e.g., definition of other users, reset of their passwords).
The IoT Devices or the IoT System requires the passing of an authentication procedure (e.g., login) before being able to allow the processing of any personal data. This authentication procedure verifies the username and a password of at least of 8 characters in length and containing alphanumeric, special, and uppercase characters.

(*Continued*)

10. ETSI, Cyber Security for Consumer Internet of Things, 2019. Document available at the following link: https://www.etsi.org/deliver/etsi_ts/103600_103699/103645/01.01.01_60/ts_103645v010101p.pdf

Table 9.1. Cybersecurity measures for an IoT device and an IoT system (Source: ETSI) (continued).

The IoT Devices or the IoT System requires strong authentication e.g., (multi-factor authentication, possession or biometrics). For IoT Devices or the IoT System that have stateless systems in general, the IoT Devices or the IoT System generates a token to associate to the session. The token associated with the session of the web IoT Devices or the IoT System or stateless systems is sufficiently long (64 or more alphanumeric characters) and impossible to guess. The token associated with the session of the IoT Devices or the IoT System or stateless systems has an expiration time.
The IoT Devices or the IoT System stores the password within its database in encrypted form.
The IoT Devices or the IoT System uses a hashing algorithm suitable for password encryption.
The IoT Devices or the IoT System implements automated password selection restrictions (e.g., a minimum number of characters is set, and it ignores common or user-referenced passwords). When the user ID is associated to an email address, the IoT Devices or the IoT System requires such email address to be verified. Email addresses associated with a user ID are periodically verified to ensure that the email is still valid and in use.
The IoT Devices or the IoT System limits or throttles the availability of logins in the event of an abnormal number of unsuccessful access attempts occurring within a short time frame.
The IoT Devices or the IoT System allows each of its administrative users to assign different permission levels to different users.
The IoT Devices or the IoT System prevents any non-administrative user from changing the permission levels assigned to other users.
The IoT Devices or the IoT System protects the data it allows to be processed through pseudonymization techniques.
The IoT Devices or the IoT System protects the data that it allows to be processed through transparent encryption techniques. Data processed through the IoT Devices or the IoT System are appropriately classified (e.g., common, particular, judicial, subdivisions in personalized under systems).
The IoT Devices or the IoT System transmits network traffic in a protected from via state-of-the-art security protocols (e.g., TLS1.2, valid certificates, HSTS). Data processed with the help of the IoT Devices or the IoT System are backed up at least daily. Data processed with the help of the IoT Devices or the IoT System can be restored quickly.
The IoT Devices or the IoT System is currently supported (e.g., through the release of security updates and patches).

(*Continued*)

Table 9.1. Cybersecurity measures for an IoT device and an IoT system (Source: ETSI) (continued).

The IoT Devices or the IoT System is constantly kept up to date. The IoT Devices or the IoT System is periodically subjected to sessions of vulnerability assessment and penetration testing to assert its robustness to cyberattacks.
The IoT Devices or the IoT System generates access logs.
The IoT Devices or the IoT System generates logs of critical actions (e.g., creation or removal of content or users).
The IoT Devices or the IoT System generates logs of the performed processes.
The logs are complete, unalterable, and stored for at least six months; the integrity of the logs can be verified. If the IoT Devices or the IoT System is connected with smartphones and requires permissions on the device, it provides policies that describe the purposes of the processing enabled by each permission. If the IoT Devices or the IoT System is connected with smartphones, it never uses the Device ID as a key to identify a record. If the IoT Devices or the IoT System is for smartphones, it uses certified pinning techniques to avoid MITM attacks.
The IoT Devices or the IoT System code does not contain confidential credential components (e.g., passwords, tokens, keys …). The IoT code is developed in accordance with the guidelines for secure code (e.g., CERT, OWASP …).

9.3.3 Data Protection by Design Measures for IoT: Recommendations

In accordance with the GDPR principles, in the Table 9.2 below, several privacy measures have been identified, in order to adequately design IoT product and systems, fulfilling the obligation under Art. 25 GDPR.

Table 9.2. Privacy measures for an IoT device and an IoT system.

The IoT Devices or the IoT System is accompanied by a specification of the type of data of which it allows the processing.
The IoT Devices or the IoT System is accompanied by a specification of the data flows from/to the outside.
The IoT Devices or the IoT System allows to define and modify the retention times for the various types of data that it stores.
The IoT Devices or the IoT System makes it possible to record the source of the data it stores (e.g., data supplied directly by the person concerned, data extracted from databases …)

(*Continued*)

Table 9.2. Privacy measures for an IoT device and an IoT system (continued).

The IoT Devices or the IoT System stores only the data necessary for its operation (e.g., it does not store unnecessary data).

If the process of verifying the accuracy of the data entered by the user identifies incorrect or suspicious data, the IoT Devices or the IoT System sends an alert to the competent function or reports it to an administrator (to allow the competent function to be informed).

The IoT Devices or the IoT System retains the date of the last update of each record.

The IoT Devices or the IoT System allows an administrator to "mark" data as restricted (e.g., providing flags in the database that identify the associated field as restricted).

Where applicable, the IoT Devices or the IoT System prevents the processing of restricted data fields (the restricted data field must not be read, modified, deleted, transmitted, displayed, etc. until it is unlocked by the platform administrator. Neither another user nor IoT Devices or the IoT System should be able to do this).

The IoT Devices or the IoT System shall enable the Data Controllers to aggregate in a comprehensible way all the data that it retains in relation to an interested party, allowing the party the ability to modify and visualize it.

The IoT Devices or the IoT System shall enable the Data Controller to record aggregated data in one or more common format files (.csv, .xlsx, .xls, .txt, etc.).

The IoT Devices or the IoT System shall enable the Data Controller to transfer aggregated data relating to a Data Subject in an interoperable format (e.g., XML, CSV, JSON).

The IoT Devices or the IoT System allows to export the aggregated data related to a Data Subject in an interoperable format (e.g., XML, CSV, JSON), flanking them with useful metadata in order to identify them correctly.

If the IoT Devices or the IoT System collects data on minors, it requires the consent of the parental guardians to be entered and given to the Data Controller.

Where applicable, if the IoT Devices or the IoT System collects data on minors, the consent of the parental guardians shall require an express opt in consent.

If the IoT Devices or the IoT System generates scores relating to a Data Subject (e.g., third-party data resulting from automatic processing) and the Data Subject did not give his or her consent to automated processing, the IoT Devices or the IoT System makes it

possible that the decision having legal effects on the Data Subject comes from an operator and not from an automated process.

(*Continued*)

Table 9.2. Privacy measures for an IoT device and an IoT system (continued).

Where applicable, the IoT Devices or the IoT System works in accordance with the consent given by the Data Subjects (e.g., it informs the operators about the consent given and does not make certain types of data available for certain processing operations if their consent was not been given).
Where applicable, the IoT Devices or the IoT System keeps a record of the consent lent or denied, each with its own timestamp.
Where applicable, the IoT Devices or the IoT System shall keep a record of the requests by the Data Subjects to exercise their rights.
The IoT Devices or the IoT System does not feed databases and/or does not transfer personal data to servers not allocated within the European Union in the absence of assessment of adequacy of the country in which the data are transferred and explicit consent requested and provided by the Data Subject/IoT Devices or the IoT System user.
If the IoT Devices or the IoT System is public, the Data Controller shall communicate and ease access to privacy policies in order to allow the Data Subject to review them at any time.

Conclusion and Acknowledgments

Considering the pervasive nature of the Internet of Things, the risk for the rights and freedoms of data subjects will be higher and higher. Adopting data protection by design approach will be key for the adoption and compliance of IoT-related services and applications.

The adoption of the GDPR has raised the level of awareness on data protection regulations. However, ensuring compliance will require to bring together experts in technology and in law in order to build a bridge between these two worlds.

The present chapter has been redacted in the context of the European Research Project NGIoT in the context of the Horizon 2020 European Research Program.

DOI: 10.1561/9781680836837.ch10

Chapter 10

Cybersecurity Certification in IoT Environments

By Sara N. Matheu and Antonio F. Skarmeta

This chapter will present and analyze current standardized IoT security certification approaches and their lacks in terms of security certification challenges, such as the security dynamicity or the life cycle management. Furthermore, the chapter will present a methodology combining security risk assessment and testing to address some of the challenges and to serve as a basis for future security certification schemes.

10.1 Introduction

The technology of the Internet of Things (IoT) is increasingly invading our daily lives. The high scope involves devices of all kinds, such as the washing machine, traffic lights, the car, the fridge, the surveillance camera, the smart band, the heart monitor, or even the clothes. These smart devices connect to the internet and exchange data in order to make our lives easier. This great interaction and exchange of information is plagued in the definition of IoT given in [1], where it is defined as "an open and comprehensive network of intelligent objects that have the capacity to auto-organize, share information, data and resources, reacting and acting in face of situations and changes in the environment." However, the advantages of the interconnectivity of these devices pose a big challenge associated in terms of security

and privacy. On the one hand, IoT devices gather a high amount of personal and sensitive data that could be exposed to attackers. On the other hand, the attack surface has been increased in such a way that a network of these devices can be used to attack more juicy targets. This is clearly seen in attacks such as the Mirai IoT Botnet [2], which causes several money loses to big companies such as Spotify or Microsoft. This network was composed by different kinds of typical devices (e.g., IP cameras, video cameras, routers, thermostats, and even computers), and although the creators were arrested, new variants of the Botnet have been developed, leveraging to a worse menace, capable of performing Distributed Denial of Service (DDoS) attacks at 1 terabyte per second. This gets still worse in situations in which human life is involved, for example, in ehealth devices or automated vehicles, as demonstrated in [3].

Being aware of this situation, the scientific community, companies, standardization bodies, and, especially, the governments try to respond to this challenge, working together to build a next generation of more secure and standardized devices. In this sense, it is needed a suitable security certification scheme to assess and compare different security technologies, in order to provide a more harmonized IoT security view to be leveraged by end consumers. The term "certification" is defined by the National Institute of Standards and Technology (NIST) as "a comprehensive assessment of the management, operational, and technical security controls in an information system, made in support of security accreditation, to determine the extent to which the controls are implemented correctly, operating as intended, and producing the desired outcome with respect to meeting the security requirements for the system." This way, the main aim of the security certification is to validate and verify the security of the product. In Europe, this initiative is led by the European Union Agency for Cybersecurity (ENISA), as established in the Proposal for a REGULATION OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL on ENISA, the "EU Cybersecurity Agency," and repealing Regulation (EU) 526/2013 and on Information and Communication Technology cybersecurity certification ("Cybersecurity Act"). As described in the Cybersecurity Act, the main objective is to create a common cybersecurity certification framework for digital products, including IoT, in a way the certificates could be valid inside the whole European Union. However, while the current heterogeneity existent in terms of security standards and certification schemes makes difficult the comparison between products certified by different standards and schemes, the dynamism associated to security leverages on an out-of-date certificate that does not show the security level in a realistic way. In this sense, agile and automated certification mechanisms are needed, implying a revolution of the current certification schemes that are usually static, time costly, and expensive.

This chapter analyses in Section 10.2 the main challenges associated to cybersecurity certification and how current certification standards deal with them. As a

response, Section 10.3 details the European Telecommunications Standards Institute (ETSI) standard [4] assessment proposal, in which our security certification proposal, described in Section 10.4, is based. The proposal aims to combine security testing and risk assessment to assess in an automated and objective way the security of an IoT device. Finally, the chapter ends by highlighting the key points discussed.

10.2 Security Certification Challenges in Current Schemes

One of the main challenges associated with security certification is the harmonization of the wide variety of security certification schemes that coexist together [5]. The current heterogeneity makes difficult the comparison of different solutions and processes, especially when a product is evaluated under different certification schemes at national levels. Currently, there is no unified solution that copes with these issues; therefore, the process of comparing and assessing the cybersecurity level of different IoT solutions is challenging. ENISA already remarked the need for harmonization of security certification that could help to increase the trustworthiness and competiveness of European products [6]. Following the recommendations of ENISA, there are some elements that should be harmonized. This is the case of the different assurance levels, the elements considered during the certification process, and the roles of the involved stakeholders. Regulatory bodies have an important role here, promoting the creation of cybersecurity framework through the consensus of the main stakeholders and orchestrating its development and deployment. In particular, the certification meta-scheme proposed by European Cyber Security Organisation (ECSO) [7] represents an ambitious initiative that homogenizes and aggregates different certification approaches under a common framework.

Another challenge is related with the standardization of the cybersecurity scheme. Despite the limitations of the current approaches, a cybersecurity certification scheme should adopt the main concepts, terms, and operational aspects of existing standard approaches [such as Common Criteria (CC) [8]]. The usage of standardized concepts helps to a common understanding and to the harmonization of the cybersecurity certification.

Existing approaches are usually expensive, slow, and complex, requiring many documentation and processes [9, 10]. This could imply that a small company could not afford the costs of the certification process or even a delay on the market release of the product, with high money losses.

Cybersecurity is a very dynamic concept. Taking into account the frequency of updates and patches of certain devices, a lightweight recertification process is necessary to ensure an updated security certificate. Automated procedures are also necessary to ensure the scalability of the (re)certification process. In this sense, the

cybersecurity certification scheme should deal with the changes of the certificate. On the one hand, the product should be monitored during its lifecycle in order to detect new vulnerabilities and update its security level. On the other hand, the security level should be modified when a cybersecurity recertification is required due to an update/patching.

Another key point is that a system composed by IoT devices could be composed by several components with different levels of security. Security composition could be a desirable design feature in cybersecurity certification and deployment at least for some security properties (e.g., confidentiality, authentication), but it is usually hard to perform in a way that all the layers and threats are correctly weighted [11]. The security certification process should also take into account the protocol stack to cover vulnerabilities at different layers. Indeed, RASEN project [12] proposes a mechanism to aggregate risks from different layers. However, physical aspects remains as a challenge, due to the context dynamicity of the IoT, which means that security properties are difficult to separate from other layers of an IoT component.

As a result of the certification process, a label should be generated to provide a simple, clear, and visual level of the security certified [13]. Companies such as Bosch [14] add that customers need to compare the security of different products without feeling overwhelmed with technical details. In this sense, the label has to address an important trade-off between the simplicity of the label and the non-ambiguous and complete representation of the results of such process. This is rather difficult, because in comparison to the energy label, which measures a physical quantity, the measurements of security are far more complex. In addition, the label design should also take into consideration the dynamicity of the security. As pointed out by ECSO WG1,[1] a visual static cybersecurity label is not enough, since it should also cope with the dynamism of the IoT paradigm to reflect changes on the current security level. For this reason, the usage of a digital QR code, which can be regenerated, can help to check the status of the cybersecurity label in a fast and easy way, as it could also be easy to update.

The context in which the product will operate must be considered, in order to make products comparable among each other and to specify the boundary conditions of the context where the cybersecurity certification was applied. This aspect is specially challenging, because it could not be known a priori.

To address the issue of label significance and the need to measure the security properties, security metrics must be established. However, some of such metrics, such as likelihood or impact, are difficult to be measured, due to its complexity, which is reported in [15].

1. https://www.ecs-org.eu/working-groups/wg1-standardisation-certification-labelling-and-supply-chain-management

While some of the previous challenges are being currently addressed by European organizations, the security dynamism of the IoT paradigm and the burden of existent schemes make the adoption of an IoT cybersecurity certification framework challenging. The next section gives a brief overview of the main current certification schemes and how they deal with these challenges. For a more detailed description, [5] poses a complete source of information of the security certification and standards.

10.2.1 Current Security Certification Schemes

The most well-known cybersecurity certification standard is CC [8], in which the security requirements are specified in Protection Profiles (PPs) for a Target of Evaluation (TOE), which is a set of software, firmware, and/or hardware. CC permits the mutual recognition between the Common Criteria Recognition Arrangement (CCRA) [16] through the definition of Collaborative Protection Profiles (cPP). CC uses Evaluation Assurance Levels (EAL) to describe numerically the depth and rigor of an evaluation. However, CC has some limitations [10, 17] such as the time and effort required to document and evaluate, in special at high EALs, or the management of changes in the certified product. This means that any change over the TOE could invalidate the result of the certification, something that is quite critical in frequent updated products.

The Commercial Product Assurance (CPA) [18] is the UK national scheme in charge of assessing the security level through a series of security characteristics published in [18]. If a product passes the CPA assessment, it is awarded with the Foundation Grade certificate, valid for two years, and allowing any type of update required, dealing in some way with the dynamicity of the security changes [18]. However, recertification has to be initiated by the manufacturer and takes about 6 months. The main barrier is that there is no Mutual Recognition Agreement (MRA), so the certificate is not recognized outside the UK.

Cybersecurity Assurance Program (UL CAP) is a certification scheme that verifies the compliance against the UL 2900 standards [19], developed by UL. However, these standards were created by a for-profit enterprise, and they were not published in order the research community could validate them. This has supposed a high amount of critics, as the standard has three parts, but only the first one has been published. Regarding recertification, in case there is a major change, the product must be certificated completely; there is no lightweight alternative process.

The Certification de Sécurité de Premier Niveau (CSPN) [20] is a French standard created by the Agence nationale de la sécurité des systèmes d'information (ANSSI) in 2008. CSPN ensures independence through the auditors, which have to be accredited by the ANSSI. The evaluation includes conformity analysis (verify

that the product complies its security specifications) and efficiency analysis (measuring the strength of the security functions and mechanisms). One of the key points of CSPN is that the evaluation is performed in a short period of time through the adaptation of the product development lifecycle, reconciling time needs of the manufacturing with the security assessment. CSPN is complementary to CC and can be used as a previous short and non-expensive assessment to CC certification [21].

The IoT Security Testing Framework [22], developed by International Computer Security Association (ICSA) Labs, specifies security testing requirements for different types of IoT systems. The approach is based on a periodic assessment and update of the certification criteria, addressing the dynamicity inherent to IoT. To strengthen the assessment process, ICSA Labs (or associated accreditation bodies) also carry out random assessments of the current product, and a short period of time (2 to 4 weeks) is given to fix any flaw encountered or the certificate may be revoked. The certificate is renewed annually by repeating the whole process. However, there is no MRA, and the criteria and processes are not standardized.

The European Privacy Seal[2] certifies if an IT product is compliant with the General Data Protection Regulation (GDPR) [23]. Other legislative documents from the Member States could be also taken into account. For the certification process, the regulatory requirements are translated into questions to be answered or addressed. However, the process is done manually by experts answering the related questions, meaning that the assessment can be affected by personal judgments. In addition, it is focused on the privacy aspect.

The Singaporean National IT Evaluation Scheme (NITES)[3] is mandatory for manufacturers that provide IT products to governmental agencies in Singapore. Although the specifications are private, it is based on a customization of CC standard v3.1 [5] at the approximate level of EAL4+ with some additions, which are mostly targeted to the vulnerability analysis. Therefore, they share similar flaws, such as the heavy documentation and the management of the security changes.

The ULD Datenschutz-Gutesiegel[4] is a German IT certification scheme used by German public authorities in order to verify the compliance with the data protection and data security rules. Although for minor changes it considers a lightweight certification process, for major changes, the overall assessment has to be repeated.

Finally, the Cellular Telecommunications Industry Association (CTIA) is a recent IoT certification scheme for LTE and/or WiFi devices [24]. The certification process takes into account security aspects such as authentication, encryption, patch

2. https://www.european-privacy-seal.eu

3. https://www.csa.gov.sg

4. https://www.datenschutzzentrum.de/guetesiegel

management, and threat monitoring. The valuation is conducted by an accredited laboratory, which executes a series of tests based on the security aspects mentioned. However, it is still too early to evaluate this scheme, since currently there are no systems or devices certified with it.

From the analysis done, there is certifiably not silver bullet security certification scheme able to cope with the challenges inherent to the IoT. IoT paradigm poses a series of challenges that current certification schemes have to address by adapting their processes, especially to deal with the security changes.

10.3 The Two Perspectives of the ETSI Approach for Security Risk Assessment

The proposed IoT cybersecurity certification framework is based on the ETSI approach described in [4]. This approach establishes two different ways to perform the security assessment based on security risk assessment and security testing. Figures 10.1 shows the two perspectives proposed on the ETSI methodology. The one on the left is the so-called test-based risk assessment perspective, and the one on the right is a risk-based testing perspective.

Both perspectives share a common initial phase, *Establishing the context*, which includes preparatory activities like *Understanding the Business and Regulatory Environment* as well as the *Requirements and Process Identification* or *Test Planning*. The first activity is meant to identify and fix the high-level security objectives. After that, the *Requirements and Process Identification* activity is focused on analyzing and

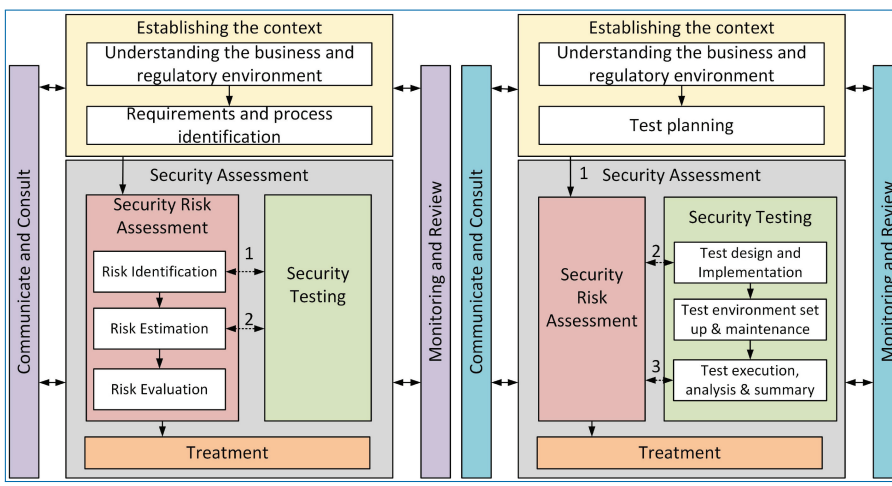


Figure 10.1. The two views of the ETSI methodology.

documenting the technical context of the TOE. In the parallel perspective, risk-based testing, the *Test Planning* activity is intended to determine the test strategy specifying the testing techniques and test phases. This perspective integrates the test planning with the risk assessment process by using security risk assessment to identify high areas of risk and optimize the testing efforts needed in consequence (see 1 in Figure 10.1, right). Moreover, the ETSI proposal points out that a previous assessment process could be useful to establish the test strategies necessary to deal with the critical risks.

The main part called *Security Assessment*, also common in both perspectives, represents the integration between the *security risk assessment* and the *security testing* work streams. In particular, it combines typical activities considered in the ISO 31000 (Risk management—Guidelines) and ISO 29119 (Software and systems engineering—Software testing) standards. The two perspectives differ in the way these two streams interact between them.

On the one hand, in the test-based risk assessment perspective, *security risk assessment* is composed by three activities. The first one, *risk identification*, is meant to find, recognize, and analyze the associated risks of the TOE. Here, the expert opinion, historical data, and stakeholder's needs can be considered. In this perspective, risk identification is improved by security testing, which is able to identify actual vulnerabilities or areas particularly vulnerable (see 1 in Figure 10.1, left). This can be done through the usage of network discovering techniques or vulnerabilities scanners. The second one, *risk estimation*, determines the risk level, understanding the origin of the risk and its consequences. However, risk estimation is usually a hard activity, as the information for the estimation is often imprecise and subjective, relying on the expert judge. This perspective tries to solve this issue by providing additional input from the security testing process with the objective to improve the confidence of the risk measurement (see 2 in Figure 10.1, left). Finally, *risk evaluation* determines if the risk obtained in the *risk estimation* activity is acceptable or not, depending on the context established at the beginning of the process.

On the other hand, and following the risk-based testing perspective, *security testing* is also composed by three activities. Firstly, *test design and implementation* is meant to design, implement, and derive the test cases that are going to be performed over the TOE. Here, security risk assessment can provide information about the potential vulnerabilities to systematically determine the test purposes. Also, the risk estimation obtained from the risk assessment can be used to prioritize the test cases (see 2 in Figure 10.1, right). *Test environment setup and maintenance* sets up the environment in which the tests are going to be executed. Finally, *test execution, analysis, and summary* executes the tests and analyses their results. One of the main challenges in this activity is the extension of the testing process taking into account the budget, the time, and the likelihood of discovering new vulnerabilities.

Especially during regression testing, which is focused on discovering security flaws after an update or patch, risk assessment can help to prioritize the execution of the tests based on their likelihood of discovering flaws (see 3 in Figure 10.1, right).

Both perspectives share an additional activity as part of the security assessment phase: the *treatment*. This activity is focused on providing countermeasures to the risk encountered in the previous processes. In this sense, four strategies can be performed. If it is possible, risk can be reduced through management controls and procedures to validate that the controls are correctly implemented to avoid the consequences of an attack. However, if the risk cannot be avoided, it can be transferred from one place to another, insuring an activity with major risks, or it can be diversified, spreading the risk to other activities in order to not loss the entire project. A more drastic action is to elude the risk by not going ahead with the project or activity that is involved in the risk. However, this may imply money loses or a major risk over a human life in case the activity is essential for it.

In parallel to the phases mentioned, the ETSI proposal also considers two activities, *communicate and consult* and *monitoring and review,* to provide the contextual and management-related information for both work streams. In particular, they aim to control, react, and improve all relevant information and results of the process.

Based on the ETSI proposal, next section details the proposed instantiation of the integration of the two perspectives to foster the development of an efficient and lightweight security certification framework.

10.4 Proposed Approach for a Cybersecurity Certification Framework

The proposed IoT cybersecurity certification framework is based on the ETSI approach described before, by combining the risk-based testing approach with the test-based risk assessment approach. As a result, risk assessment metrics can be obtained from testing results in an objective way, and risk assessment can be used to prioritize the implementation and execution of the security tests. In addition, we added additional aspects inherent to the security certification, such as the labeling process. This process is further considered by regulatory and security organizations, and it is explicitly mentioned in the Cybersecurity Act [7, 25]. The label obtained as a result of the labeling is meant to show in a visual way the result of the certification process in a way a non-expert consumer could understand it. To homogenize the terms, we refer to the objective of the evaluation as TOE, following the same definition of CC. Nevertheless, our approach also considers in the definition a specific configuration (i.e., cryptographic suite and security parameters, libraries, protocols, etc.) and the context in which it operates. Figure 10.2 shows an

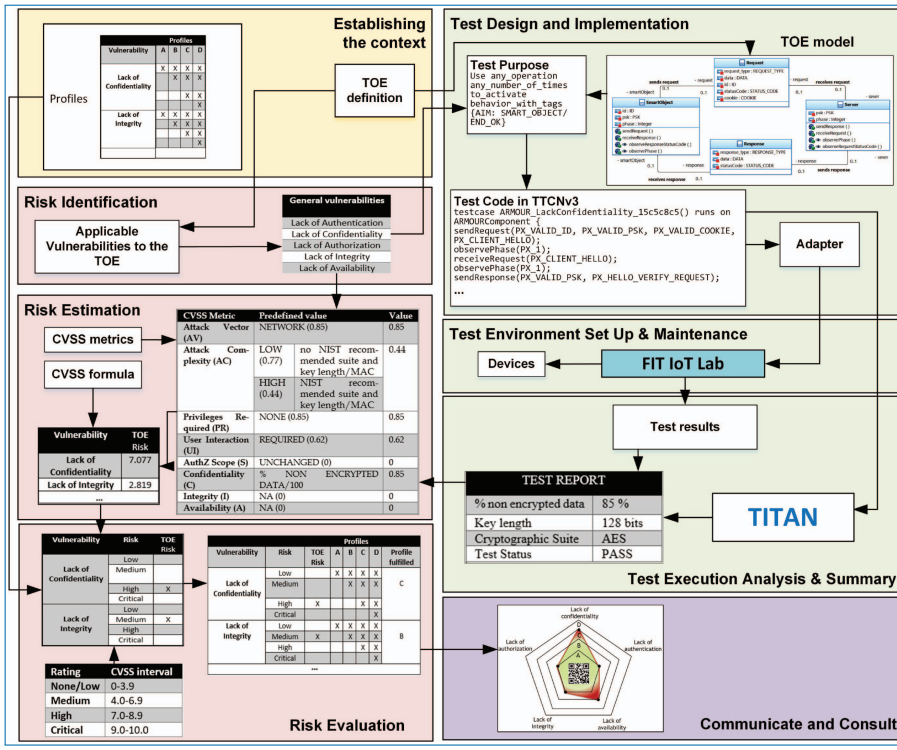


Figure 10.2. Proposal instantiation with an example.

example of the proposal instantiation. The different steps of the figure are going to be described in the next subsections to clarify the insights of the process. For more detailed examples, the reader can refer to [26–28].

As a starting point, the certification approach considers a set of five general security properties obtained from the literature, in which more specific vulnerabilities and threats are mapped (e.g., from existing vulnerabilities databases such as the National Vulnerability Database (NVD)) [32]:

- Lack of authentication. The endpoints should be legitimate.
- Lack of integrity. Received data is not modified during the communication; if data is modified, it should be detected.
- Lack of confidentiality. Transmitted data only should be readable by the legitimate endpoint.
- Lack of authorization. Services should be accessible only to endpoints who have the right to access them.
- Lack of availability. Exceptions and errors should be controlled to avoid faults that affect the system.

It should be noted that the proposed mapping is designed to measure the risk associated with the lack of each property, so the resulting security label can be specified by using such properties to provide a multidimensional security description.

10.4.1 Establishing the Context

This phase is in charge of gathering all the information related with the context of the TOE, in a way it could be reflected during the security assessment. In the first activity, named *Understanding the Business and Regulatory environment*, the focus is to determine which level of security is required in a particular domain. To this end, it requires a deep analysis of the laws and the regulatory environment. Experts from the different domains could be required to give a common comprehension of the laws and regulation environment.

Once the regulatory and business requirements of each domain have been analyzed, the second activity, *Requirements and Process Identification*, is in charge of defining a set of profiles (A. B, C, and D) to reflect the different security levels (low, medium, high) for a specific domain, in a similar way to the energy labels. The profiles are meant to indicate the risk acceptable associated to each security property to obtain a profile. For example, if the profile A for home building needs a lack confidentiality low, only a TOE with a low risk in confidentiality is going to be able to obtain the profile A. It is worth noting that if the TOE fulfils a specific profile, it will fulfill also the lower profiles, that is, higher profiles indicate higher requirements in the security. Figure 10.2 shows an excerpt of a set of profiles defined for a specific domain. In the example, the profile A requires a low risk in confidentiality, whereas in profile D a high risk is acceptable.

The third activity, *Test Planning*, is in charge of analyzing the TOE security requirements and to design a testing strategy according to it. This activity could be also used to prioritize the tests during a regression testing phase. Additionally, the establishing the context process interacts with the additional *Communicate and Consult* and *Monitoring and Review* processes. The objective of this interaction is to keep updated the profiles according to the expert's judgment, the current regulation, and the encountered vulnerabilities.

10.4.2 Security Testing

In contrast to the ETSI proposal, the instantiation of this phase is intended to use specific methodologies and tools for test generation and execution with the objective of automating as much as possible the security testing. At the same time, the automation of security testing aims to link the security updates with the security label, coping with the changing conditions of the TOE and reflecting the real security level in any moment.

The first activity, *Test design and implementation*, designs a test suite to test the risk of each vulnerability. With the objective of automating this process, we use a Model Based Testing (MBT), which generates the tests from a high level model of the TOE [29]. Indeed, MBT has shown its benefits and usefulness for systematic compliance testing of systems [30]. On the one hand, our approach uses the Unified Modeling Language (UML) class diagrams to model the architecture of the TOE, that is, the entities involved and their relations. Figure 10.2 shows an example of an architecture modeled in UML. Specifically, the figure shows two main entities (smart object and server) that are connected with other two entities (request and response), by the relations "receiveRequest", "sendRequest", "receiveResponse," and "sendResponse". On the other hand, the TOE behavior is expressed in Object Constraint Language (OCL),[5] using the CertifyIt tool [31]. It is worth noting that other tools can be used to automate this process [32]. The high-level tests are defined in CertifyIt by means of test purposes, referred to the high level model. Figure 10.2 shows an example of test purpose, in which any operation can be used any number of times with the objective of reaching a specific tag called "smart_object_ends_ok." The power here resides on the tags we can specify in OCL and the ability of CertifyIt for generating all the steps of the test necessary to reach the tag. This way, it is not necessary to specify step by step the test, implying less time to program and more time to think about security.

The generated tests can be exported in several languages (XML, PDF, JUnit, and TTCN3). In particular, we export the tests in Testing and Test Control Notation (TTCN) v.3 language, as it is a standardized testing language whose aim is to systematically execute the test, improving efficiency and scalability. Figure 10.2 shows an excerpt of a test exported in TTCN-3 language. To cope with the particularities of each IoT device, CertifyIt also produces some interfaces, called adapter, as a middle interface between the TTCN-3 and the real device. The adapter has to be implemented to link the operations and the types defined in the high-level model with the real code.

In the *Test environment set up and maintenance* activity, the execution environment is set up (e.g., a local device or a large-scale platform). Our proposal considers the usage of the FIT IoT-LAB platform, a large-scale infrastructure (about 2000 IoT nodes) for testing purposes without the need of cumbersome deployment tasks [33]. In this case, the setup of the environment consists of the reservation of the resources and the upload of the code to the remote devices.

Finally, the tests are executed in the *Test execution, analysis, and summary* activity. To do so, we use the tool TITAN. TITAN is a TTCN-3 compilation and execution environment for different platforms that in combination with CertifyIt creates

5. http://www.omg.org/spec/OCL/2.4

executable tests. TITAN sends the test commands (TTCN3 language) to the real device through the adapter in order to execute the tests. As a result of this process, it generates a test report indicating the tests that have passed or failed as well as additional information (e.g., time, number of devices, errors, sniffer information).

The automation of the testing process implies that if a new vulnerability is discovered, the recertification process can be done in a non-expensive, fast, and easy way, which is key to address the dynamic nature of cybersecurity in IoT.

It is worth noting that the proposed tools for the automation are independent of the methodology used to automate the process. This chapter proposed a particular combination but other one is also possible.

10.4.3 Security Risk Assessment

Security risk assessment aims to measure the risk using the results of the testing process in order to obtain a numerical value that could be used to compare the security obtained by different devices. The obtained risk value is used to select the profile fulfilled by the TOE, defined in the establishing the context phase.

The first activity of the Security risk assessment phase, Risk identification, analyzes and selects according to the TOE the applicable vulnerabilities that should be tested. For example, if the scenario considered does not interact with resources, and therefore, authorization does not make sense, it is not considered. These non-applicable general vulnerabilities are labeled with a low risk by default, as the vulnerability cannot be exploited. The vulnerabilities are mapped to the five general vulnerabilities enumerated at the beginning of the section to aggregate their risk marks after the individual risk calculation. Finally, the selected vulnerabilities are used as input for the testing process, to define the test purposes and model the necessary elements in the TOE model.

After the security testing phase, the test report is used in the risk estimation activity, to obtain a risk measure for each vulnerability considered. We use the Common Vulnerability Scoring System (CVSS) [34] to perform the risk estimation. CVSS consists of three metric groups: Base, Temporal, and Environmental. The Base group represents the intrinsic qualities of a vulnerability, the Temporal group reflects the characteristics of a vulnerability that change over time, and the Environmental group represents the characteristics of a vulnerability that are unique to a user's environment, dealing with the context influence challenge. The Base metrics produce a score ranging from 0.0 to 10.0, which can be modified by scoring the optional Temporal and Environmental metrics (they include a metric value that has no effect on the score). The CVSS metrics to measure the risk of each vulnerability are obtained from the test results. This way, the risk is measured in an objective and empirical way. Figure 10.2 shows an example of how the test

results are mapped into CVSS metrics. There, the percentage of non-ciphered data is assigned to the attack vector, and the complexity of the attack is based on the ciphersuite used (algorithm and key length). The metrics are combined using the CVSS formula [34].

Finally, the risk evaluation activity compares the obtained risk in each general vulnerability with the profiles available for a specific context. To do so, the numerical risk is mapped to a risk interval (low, medium, high and critical), and the profile fulfilled is obtained by comparing the risk interval with the risk interval of the profile. Figure 10.3 shows the process of mapping the risk intervals and comparing the result with the profiles (A, B, C, and D). In this case, for the lack of confidentiality, the TOE obtained a high risk, which fulfils profiles C and D. As the chosen profile is always the highest one, the profile obtained for that general vulnerability is C. This process is repeated for each general vulnerability.

10.4.4 Communicate and Consult: Labeling

One of the main results of the security certification process is the obtaining of a security label. We considered the *labeling* phase as part of the *communicate and consult* phase described in the ETSI proposal. The label has to take into account the context in which the TOE will be deployed, the security level achieved, and how the security has been assessed. For the last one, we propose the EALs specified in CC, from EAL1 (functionally tested) to EAL7 (formally verified, design, and tested).

To provide a simple label that could be understood by non-expert consumers and that could be able to transmit enough information about the security obtained, we propose a spider chart diagram as a label (a pentagon). The label has the five general vulnerabilities as vertex and the profiles as edges. This way, the intersection of the vulnerabilities with the profiles shows the security level achieved for each vulnerability. Figure 10.2 shows an example of label. Lack of confidentiality has the profile C, lack of authorization, authentication and integrity has the profile A, and lack of availability has the profile B. As the profiles are organized from the lower to the higher, a major area means more risk, facilitating through a visual concept the comprehension of the label.

In addition, and following the ECSO recommendations, we propose the usage of a QR code to update the label after a recertification process, coping with the security dynamism.

10.4.5 Monitoring and Review

Although the Monitoring and review activity is not addressed in our proposal, certain mechanisms could be used to manage the security changes of the TOE.

One of the main sources of information to detect a security change is a vulnerability database. Monitoring the activity of the databases and analyzing if the vulnerabilities added are applicable to the TOE could be used to initiate a recertification process. Updates and patches are also mechanisms that could imply a security change and should be monitored.

10.5 Conclusion

This chapter has discussed the main challenges that have to be addressed towards the creation of a cybersecurity certification framework, and how current certification schemes fail to deal with some of them. As a response, we have proposed a methodology based on the ETSI proposal that combines risk assessment and testing to assess IoT devices in an objective and automated way, coping with one of the main challenges: the security dynamism. The methodology proposed is intended to serve as a basis for the development of a common security certification scheme, with the aim of supporting the implementation of the Cybersecurity Act.

Acknowledgments

This work has been partially funded by the European Union's Horizon 2020 research and innovation program under grant agreement no-779852 (IoTCrawler) projects and grant agreement no-830929 (CyberSec4Europe) by the FPU-16/03305 research contract of the Ministry of Education and Professional Training of Spain.

References

[1] S. Madakam, R. Ramaswamy, and S. Tripathi, "Internet of Things (IoT): A Literature Review", *JCC*, vol. 03, no. 05, pp. 164–173, 2015, doi: 10.4236/jcc.2015.35021.

[2] C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas, "DDoS in the IoT: Mirai and Other Botnets", *Computer*, vol. 50, no. 7, pp. 80–84, 2017, doi: 10.1109/MC.2017.201.

[3] D. C. Miller and C. Valasek, "Remote Exploitation of an Unaltered Passenger Vehicle", p. 91, 2015.

[4] ETSI, "ETSI EG 203 251: Methods for Testing & Specification; Risk-based Security Assessment and Testing Methodologies". 2015.

[5] ECSO, "State of the Art Syllabus v2". 2017. [Online] Available in: https://ecs-org.eu/documents/publications/5a31129ea8e97.pdf
[6] H. Baars, R. Lassche, R. Massink, and H. Pille, "Smart grid security certification in Europe. Challenges and recommendations". ENISA, 2014.
[7] ECSO, "A Meta-Scheme Approach v1.0". 2017.
[8] CCRA, "Common Criteria for Information Technology Security Evaluation. Part 1: Introduction and general model." 2017.
[9] S. Murdoch, M. Bond, and R. J. Anderson, "How Certification Systems Fail: Lessons from the Ware Report", *IEEE Security & Privacy Magazine*, vol. 10, no. 6, pp. 1–1, 2012, doi: 10.1109/MSP.2012.89.
[10] S. P. Kaluvuri, M. Bezzi, and Y. Roudier, "A Quantitative Analysis of Common Criteria Certification Practice", en *Trust, Privacy, and Security in Digital Business*, vol. 8647, Cham: Springer International Publishing, 2014, pp. 132–143.
[11] M. Bartoletti, P. Degano, and G. L. Ferrari, "Security Issues in Service Composition", en *Formal Methods for Open Object-Based Distributed Systems*, Berlin, Heidelberg, 2006, vol. 4037, pp. 1–16, doi: 10.1007/11768869_1.
[12] RASEN project, "D3.2.3. Techniques for Compositional Test-Based Security Risk Assessment v.3". 2015.
[13] AIOTI, "Report on Workshop on Security and Privacy in the Hyper-Connected World". 2016.
[14] J. Hubner and M. Lastovka, "BOSCH Political Viewpoint. Security in IoT." 2017.
[15] J. Hearn, "Does the common criteria paradigm have a future?", *IEEE Security & Privacy Magazine*, vol. 2, no. 1, pp. 64–65, ene. 2004, doi: 10.1109/MSECP.2004.1264857.
[16] Common Criteria, "Arrangement on the Recognition of Common Criteria Certificates In the field of Information Technology Security". 2014.
[17] F. Keblawi and D. Sullivan, "Applying the common criteria in systems engineering", *IEEE Security & Privacy Magazine*, vol. 4, no. 2, pp. 50–55, Mar. 2006, doi: 10.1109/MSP.2006.35.
[18] CESG, "The Commercial Product Assurance (CPA) build standard". 2014.
[19] Underwriters Laboratories, "UL 2900 Standards Process", *UL 2900 Standards Process*. [Online]. Available in: https://industries.ul.com/cybersecurity/ul-2900-standards-process.
[20] ANSSI, "Certification de sécurité de premier niveau (CSPN)", *ANSSI*, 2008. [Online]. Available in: https://www.ssi.gouv.fr/administration/produits-certifies/cspn/.
[21] G. Baldini, G. Giannopoulos, and A. Lazari, "Annex 8: JRC Analysis and recommendations for a European certification and labelling framework for cybersecurity in Europe", European Commission, 2017.

[22] ICSA, *ICSA Labs IoT Security and Privacy*. 2016.
[23] European Parliament, "REGULATION (EU) 2016/679 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation)", 2016. [Online]. Available in: https://eur-lex.europa.eu/eli/reg/2016/679/oj.
[24] CTIA, "Cybersecurity Certification Test Plan for IoT Devices". 2018.
[25] European Parliament, "REGULATION (EU) 2019/881 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 17 April 2019 on ENISA (the European Union Agency for Cybersecurity) and on information and communications technology cybersecurity certification (Cybersecurity Act)". 2019.
[26] S. N. Matheu, J. L. Hernandez-Ramos, and A. F. Skarmeta, "Toward a Cybersecurity Certification Framework for the Internet of Things", *IEEE Security Privacy*, vol. 17, no. 3, pp. 66–76, May 2019, doi: 10/gf256z.
[27] S. N. Matheu, S. Perez, Hernandez-Ramos, and A. F. Skarmeta, "On the automation of security testing for IoT constrained scenarios", en *20th World Conference on Information Security Applications (WISA)*, Jeju, Korea, 2019.
[28] S. N. Matheu-Garcia, J. L. Hernandez-Ramos, A. F. Skarmeta, and G. Baldini, "Risk-based automated assessment and testing for the cybersecurity certification and labelling of IoT devices", *Computer Standards & Interfaces*, vol. 62, pp. 64–83, Feb. 2019, doi: 10.1016/j.csi.2018.08.003.
[29] F. Bouquet, C. Grandpierre, B. Legeard, F. Peureux, N. Vacelet, and M. Utting, "A subset of precise UML for model-based testing", en *Proceedings of the 3rd international workshop on Advances in model-based testing – A-MOST '07*, London, United Kingdom, 2007, pp. 95–104, doi: 10.1145/1291535.1291545.
[30] G. Bernabeu, E. Jaffuel, B. Legeard, and F. Peureux, "MBT for global platform compliance testing: Experience report and lessons learned", en *25th IEEE International Symposium on Software Reliability Engineering Workshops*, Naples, Italy, 2014, doi: 10.1109/ISSREW.2014.91.
[31] B. Legeard and A. Bouzy, "Smartesting CertifyIt: Model-Based Testing for Enterprise IT", en *2013 IEEE Sixth International Conference on Software Testing, Verification and Validation*, Luxembourg, Luxembourg, 2013, pp. 391–397, doi: 10.1109/ICST.2013.55.

[32] W. Li, F. Le Gall, and N. Spaseski, "A Survey on Model-Based Testing Tools for Test Case Generation", en *Tools and Methods of Program Analysis*, Cham, 2018, vol. 779, pp. 77–89, doi: 10.1007/978-3-319-71734-0_7.

[33] "FIT/IoT-LAB – Very large scale open wireless sensor network testbed". [Online]. Available in: https://www.iot-lab.info/.

[34] FIRST, "Common Vulnerability Score System (CVSS) v3", 2015. [Online]. Available in: https://www.first.org/cvss/v3.1/specification-document.

DOI: 10.1561/9781680836837.ch11

Chapter 11

Firmware Software Analysis at Source Code and Binary Levels

By Franck Vedrine, Florent Kirchner, Basile Starynkevitch, Andrea Battaglia, Mario Villiani and Konstantinos Loupos

The CHARIOT design method is oriented to bridge the systems engineering gap that currently exists between (a) the formal safety engineering techniques applied in the development and testing of safety critical systems and (b) the rapidly evolving and ad hoc manner in IoT devices are developed and deployed. In this direction, CHARIOT started with a classification and usage guidelines of relevant standards and platforms to align the project developments with existing guidelines, standards, and industrial challenges on Industrial IoT Security. A free software compilation toolset is also being developed (based on existing open source technologies) targeted towards IoT engineers designing IoT systems and developing source code running on devices in Industrial IoT networks. This introduces new methods and tools for more secure and safer IoT software development with static source code analyses to help avoid safety and security defects and risks targeting the firmware developer during the building steps and the software security engineer before the firmware deployment. Source code analyses are executed offline and are strongly bound with the CHARIOT Security Engine to online check results. For that purpose, the static analyses add meta-data (cryptographic signature, static analysis results, and support for "proof-carrying code" approach [1]) into the binary, hence permitting that binary executable to be suitably "filtered" or "authenticated" by the CHARIOT

Security Engine and the IoT devices and, in turn, shielding against cyberattacks at source code (compilation and run-time) level.

At the same time, CHARIOT recognizes security vulnerabilities present at devices producers' level underestimating the potential risks to which they are exposed to, since the latter are connected almost continuously to the internet. Great risks are created by superficial behavior conducted by the manufacturers themselves who are oriented to release the firmware updates with an excessively low frequency and sometimes, fortunately only in extreme cases so far, the updates are not released at all. Existing challenges related to the actual firmware running in IoT devices include reverse-engineering of the entire firmware, extracting the file system and understanding how the entire device works, including knowing the possible use of known-to-be-vulnerable out-of-date API/libraries or unknown exploitable vulnerabilities in the firmware code itself. They also include backdoor insertion, change of the device behavior, altering its performance, without altering the telemetry values reported to administrators, finding hard-coded private symmetric-cryptography keys/passwords/user-names or private certificates used to encrypt communications between the device and other systems and rolling back the firmware to a previous legitimate version with known vulnerabilities (to be exploited). This attack is particularly insidious because the pushed firmware is authentic, so it can easily survive most of the in-place controls, as usually, they tend to check just the firmware source and/or the firmware integrity.

CHARIOT introduces a firmware checking approach comparing the individual functions that make up the firmware, looking for similarities between the functions themselves (according to certain parameters to be extracted and analyzed) and the behavior of the firmware according to statistical surveys. The firmware update process is also affirming compilation level warnings (metadata), firmware hashing and versioning through the blockchain system, as well as the firmware analysis results.

The CHARIOT firmware checking is multiple from the development of the first firmware components to the deployment and the upload of the firmware at the level of sensors, gateways, and other nodes. To prepare an industrial deployment, some verification steps are optional and depend on the Security Policy of the Industrial. The mandatory firmware verification concerns the one performed by the Security Engine in the Fog Node, just before the firmware deployment. Hence, the Security Engine has the responsibility of the firmware deployment on the sensors, gateways, and other nodes. Since the Security Engine has few times to take a decision for the acceptance of the reject of an update, the firmware needs to exhibit detailed information about its content. The different verification steps are registered in the blockchain that gives insurance about the origin and the integrity of the verification process.

The CHARIOT architecture is based on the deployment of Fog Nodes that are the entry point of all the communications to the sensors, gateways, and other

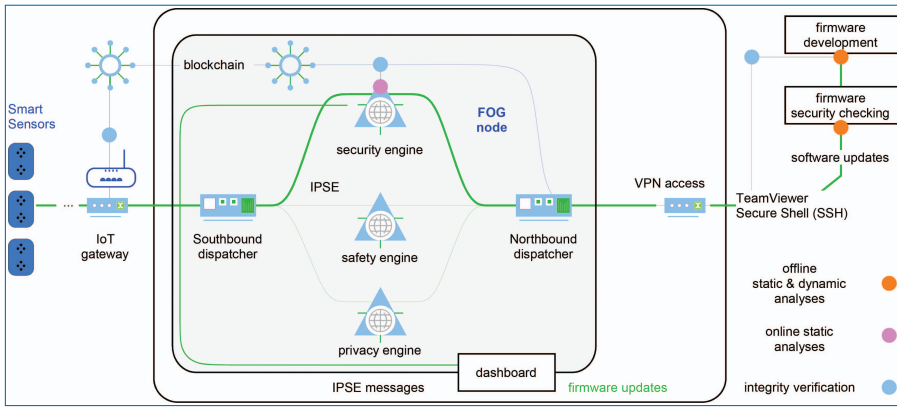


Figure 11.1. Firmware development and deployment (in green) in the CHARIOT architecture.

notes. The Figure 11.1 defines the firmware development and deployment path on the architecture of CHARIOT.

11.1 Scope, Business Orientation, and Purpose

The cybersecurity challenges relating to IoT firmware development and deployment relate to the application of security rules on the firmware of IoT devices during its source code development as well as at the binary (executable) levels. This is directly aligned to the, on-purpose or not, vulnerabilities commonly (and recently) found in IoT devices including pre-installed firmware. For instance, this could prevent sending confidential data to public addresses. The Static Analysis technologies will also reduce the number of vulnerabilities present in IoT Software like the absence of Run-time Execution errors (such as memory overflows) that are a target of the CHARIOT Static Analysis Component. Automated source code analysis techniques provide significant advantages as compared to current state of practice in IoT involving firmware source code manual inspection, while the current state of the art in critical embedded systems is the application of formal methods.

The source code analysis with static analyzers is targeting specific security rules (increased verification with decreased human resources) providing formal guarantees (at the firmware level) validating thus the developed firmware through its whole cycle, from development to deployment, through embedding of static analysis results (warnings during code compilation) in the binary of the device (under update). CHARIOT also provides an extra protection by affirming the specific source code hashing (created during the compilation time) through blockchain and thus eliminate risks even from attackers having access to the firmware development and attach meta-data (analysis results) claiming a safe firmware although it includes

introduced backdoors. A blockchain-enabled, firmware versioning system is also enhancing the update process robustness.

Automation of the registration of the analysis results and their evolution over the full development process (BISMON scope) removes human involvement needs in removing all warnings issued by the analysis as currently done at the end of the development process. By doing it incrementally in connection with the developer, it is expected to increase speed and precision while decreasing development costs. Increased Integrity (link between the source code and the running binary firmware) is ensured via analysis done within the compiler that generates the final code.

For industrials that have not access to the firmware source code or that review the source code at the end of the development process, CHARIOT offers an alternative to add some analysis results in the firmware meta-data. This alternative process should motivate firmware suppliers to do more analysis and to add certification in their firmware.

At the same time, at binary (executable) level, CHARIOT provides a solution of processing firmware binaries and identifying security vulnerabilities (e.g., code injection, jumps inside the code) by cross-referencing historical software updates and vulnerability databases as an execution layer affirmation. The presence of adequate meta-data in the firmware is considered to help in the decision of firmware update approval/rejection.

11.2 Technological Innovation and Security Alignment Per Outcome

11.2.1 Securing Firmware Through Rule-based Code Analysis and Injection of Analysis Results and the Source Code Hash Within the Binary Code

Syntactic-rule-based code analysis supports improvement of the security of firmware during the development process and during the firmware update through embedding of the analysis results, a firmware hashing and version in the firmware binary (as meta-data). At the same time, the produced source code hashing and version are included into the blockchain system (for later verification during the firmware update process). Security properties targeted by the static analyses include Rule-based Security properties and minimization of common vulnerabilities based on Run-time Errors (memory overflows, stack overflows, index overflows). These contribute to the global security of the system. CHARIOT also provides means for preventing a malicious firmware developer from adding a backdoor and attaching artificial meta-data (plus a correct hashing) to hide his backdoor. For this purpose,

the crucial analysis results will be checked by the Security Engine without any assumptions on their origin (inspired from proof-carrying code approaches).

11.2.2 Securing Firmware During the Development Process

For the firmware's developer, CHARIOT has developed an open source tool BISMON [2] that follows the development process to incrementally register the analysis results as soon as they are available. BISMON contains a document-/graph-based database and a server that centralizes the Static Analysis Result with the code. As depicted on the Figure 11.2, BISMON comes with GCC plug-ins that do the static analysis during the compilation of the source files/translation units. This ensures that the object code produced by the source files is completely consistent with the analysis results—this is ensured by the compiler. Another interest is that the analysis does not require a specific build process. It just requires some additional options to the compilation process, which are easy to add in existing makefiles.

The compiler scans the source code only once to produce the object code. That has a consequence on the BISMON analysis done during the compilation: the analysis' messages at this step only concern intra-procedural analysis and are based on syntactic rules. The plug-ins then send the analysis results and the code to the BISMON server that will perform additional verification since it has more caller/callee information about the functions in the code—hence, the inter-procedural part of the analysis is done on the server's side. To implement this inter-procedural analysis, BISMON provides a Domain-specific Language that is compiled into algorithms in the server. These algorithms follow the call-graph and consolidate the analysis

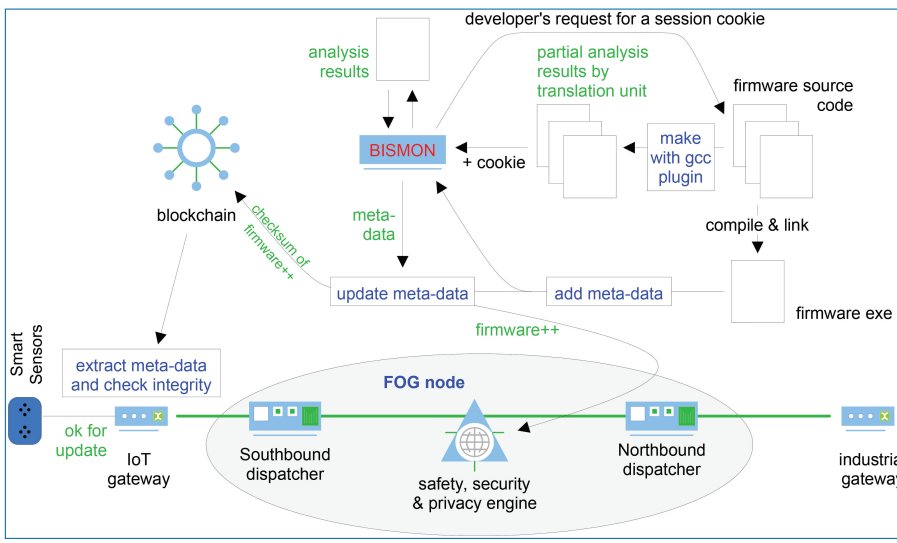


Figure 11.2. Static verification during the development process—detailed process.

results with caller/callee information. All this work operates incrementally on the analysis data provided by the developer.

We present now an example of BISMON usage. It concerns both safety and security. To ensure some integrity for the analysis results, the developer needs to be identified in the BISMON server as explained in Figure 11.2. Then, it launches the compilation command, which produces the analysis results and send them to BISMON. Let us see how it works on for an analysis that computes the maximal size of the stack usage during the firmware execution. For Safety and Security reasons, when the firmware runs, it should guarantee that the stack memory that it uses will not overflow over the available memory. This can be ensured by a static analysis at source code level. During the compilation process, the BISMON analysis computes the maximal stack frame of every function. Such an analysis is only possible in the compiler that produces the final binary firmware since the compiler knows these data—it needs them to call and return from functions (update of the stack frame registers). In the gcc plug-in = BISMON client, it is implemented as a gcc low-level pass. The BISMON analysis also produces the couples caller/callee to be able to produce the complete call graph. Then, BISMON sends these data: maximal stack frame by function and the couples caller/callee to the BISMON server. At the server side, BISMON detects the absence of recursive functions that could lead to an infinite stack usage. It transforms the call graph into a call tree by "inlining" the function calls at the caller site. It then incrementally update the call tree each time the developer sends a new translation unit. When the developer operates the link edition that generates the final executable, it sends this information to the BISMON server. At this point, the server knows that it has the entire source code. So it computes an overestimate of the stack height for every leaf of the call tree and defines the maximal value as the result for the stack usage. It then converts this result into CHARIOT meta-data and attaches them at the end of the binary firmware—in the CHARIOT meta-data section. It informs the blockchain of its work. Later, the firmware will be uploaded: the Security Engine can look if the meta-data are present and if the stack usage is bounded. If it is the case, it authorizes the sending to the CHARIOT Gateway. The gateway extracts the meta-data from the firmware: it knows the sensors to update; so if the stack usage is greater than the available memory it will not send it for the update; instead, it informs the CHARIOT Fog node that the update has not occurred.

BISMON can automatically verify (safety and security) coding rules (some buffer overflows, Insecure random numbers, etc.) and identify the lines of source code that break a particular coding rule (no recursive function, no stack overflow) as safety properties. As an example, BISMON security properties can check that every call to a communication library should have URL addresses that are statically well known with identified corporate prefixes. Such coding rules might prevent some developer's mistakes but it will not prevent an attacker from introducing

malicious code – this problem is more the scope of existing formal methods (e.g., axiomatic semantics, operational semantics, and abstract interpretation) developed in the next section.

➢ **Securing firmware before the deployment process**

In the CHARIOT approach, additional specific static analyses embed semantic and formal analyses (to check the absence of memory overflow, of Cross-site scripting). If a formal analysis generates no alarm, then the firmware execution cannot expose the vulnerability. Such type of analysis is conducting by a Security Engineer that inspects the source code, the binary code to look for security vulnerability. Formal analysis has also the advantage that their results are objective. Hence, the Security Engineer does not need to sign that his analysis is correct, since the results can be checked again by the Security Engine.

Consider the example of memory overflow: In some cases, they create security breaches since an attacker can exploit a memory overflow to take the control of the machine by executing own (external) code with an overflow of one or two bytes, jumping between codes (code injection vulnerability). If a formal analysis generates some alarms, the Security Engineer will perform different actions depending if the alarm is a true one or a false one. For a true alarm, he will ask or do a code correction; For a false alarm, he will write/generate a formal reasoning that complements the analysis' reasoning and so removes the false alarm for the security engine. At the end of this process, the running system should not have any vulnerability targeted by the analysis. The natural vulnerabilities targeted are Run-time Execution Errors (memory overflows, integer overflows, invalid pointers, undefined behaviors). Security involves other risks and new software vulnerabilities will be investigated (tainting, impact analysis) in accordance with the Security policies of industrial IoT players.

Some tools like Frama-C [3] can perform such formal analysis for IoT code [4]. Current development work in CHARIOT concerns a way to save such formal results in the firmware meta-data to check these results again with the Security Engine. It addresses formal source code verification and formal binary code verification, depending on the presence on source code or not. It is an innovative challenge in both cases: the link between the source code and the binary code needs to be established for source code verification; the formal analysis at binary code needs to recover the data structures, the control, and data flows.

➢ **Security Engine: processes firmware binaries and identifies security vulnerabilities (e.g., code injection) by cross-referencing historical software updates and vulnerability databases**

The Security Engine aims to keep the firmware (binary) of IoT devices secure through the CHARIOT environment as an important attack vector and a highly

vulnerable point of the technological infrastructure (IoT Network). Its scope is to maintain them updated/secure considering threats and exploitations that could tamper their security, compromising the functionality and the safety of the entire IoT system (and devices).

The Security Engine service scans binary files of the pre-OS (firmware) of new releases of firmware devices against older, valid versions before they get flashed on target devices (IoT devices and gateways). The Engine will check if newer versions may have unexpected behavior leading to security issues. This is achieved using a heuristic approach by observing behavioral variations from a "reference" version of the firmware, by analyzing the instruction pattern and calculating their change within the different versions. This feature is combined with preventive researching of vulnerabilities (like arbitrary Memory Access, Buffer Overflows, Format String Vulnerabilities) by searching code signatures for the target architecture that may lead to security issues.

This solution makes the Security Engine an innovative automated mechanism for checking and updating IoT devices firmware able to provide attack surface minimization, decreasing all threats that could be used in a remote way by a potential hacker, such as:

- Reverse engineer the entire firmware, extract the file system, and understand how it would be used for known-to-be-vulnerable out-of-date API/libraries or unknown exploitable vulnerabilities in the firmware code itself;
- Insert a firmware backdoor, making the device controllable by remote;
- Change the device behavior, altering its performance and the telemetry values;
- Find hard-coded private symmetric-cryptography keys/passwords/user names or private certificates used to encrypt communications between the device and other systems; the attacker may be able to eavesdrop these private communications.

➢ **Security Engine: filter firmware based on rules applied on the injected analysis results**

During the software update process, the security engine extracts the static code analysis results (meta-data) from the firmware binary as inserted earlier at the binary compilation stage. The extracted static code analysis results are then cross-referenced with a local security policy that dictates the tolerated static code analysis limits, and if the firmware passes these limits, the binary analysis process is initiated. During binary analysis, insight to the internal workings of the firmware is provided by comparing jump addresses with historical firmware update data, when available, and databases of common vulnerability jump patterns. This binary analysis

is then consumed by the security engine to approve or reject the firmware update and provide contextual messages to the initiator of the update, such as the type of vulnerability pattern identified. The overall workflow achieves two things: firstly, it prohibits anyone to issue a firmware update that possesses warnings that were ignored from the static code analysis if they do not satisfy the criteria set forth in the local security policy; and secondly, it prevents any firmware update of going through if a vulnerability or erratic pattern in comparison to its previous versions is identified in the jump registers of the binary file.

➢ **Comparing firmware by heuristic method approach**

The heuristic approach first treats the system as different sub-systems so that the sub-system's solution must spread widely at the solution space. Similarly, following this approach, the analysis was addressed by considering different aspects of the characteristics of the firmware and its typical behavior.

So, the optimal sub-system solutions in the heuristic approach for CHARIOT firmware devices are found:

- In a first stage, by comparing the individual functions that make up the firmware;
- In a second stage, going to look for a similarity between the functions themselves, according to certain parameters to be extracted and analyzed;
- After that, as the further sub-system solution, the behavior of the firmware according to a statistical survey will be taken into consideration.

Finally, the optimal solution will be determined by matching all the sub-system solutions, calculating a final reference for the statement of security of the new firmware, finding the global optimal solution more efficiently.

First phase is the Data Collection; the scope of this stage is to parse the ELF, BIN, and HEX firmware images using RADARE2 opensource tool to load images, collect data, parse an image with tool analyzer in order to get names and addresses for functions, build a map of function names and extract the CFG (Control Flow Graph), and perform initials automatic analysis.

Once each firmware function is extracted (together with basic information like addresses, names and sizes), each function is disassembled to:

- Get all the instructions used inside the function, classify the commands based on their behavior, get conditional/unconditional jumps, loops, etc.;
- Count the number of each type of instructions (standard and classified), jumps, etc.;
- Note offsets of commands.

In the second phase, the scope is to analyze the function structures taking care marginally the order of instructions inside each function, as well as the offsets, parsing the divided code in a function-by-function manner without any correlation. Then, a fingerprint sha256 for each function (a sort of UID) has been calculated, so that we are able to make a comparison between functions by matching their fingerprint.

If they match (independently from their correlation), there would be a reasonable confidence assurement for the possibility of a complete correspondence between them. A function-by-function match allows the algorithm to map all the instructions inside the two examined firmwares and perform an effective comparison.

In addition, in order to give greater reliability to the algorithm, fingerprints that take into consideration offsets have been created. This can be added as a separate parameter in the implementation. Multiple options can be considered as contributions to finger print: statistics, instructions' order, and offsets. Obviously, these options influence the final comparison results. Some functions may have the same number of instructions but a different order. Some functions may be considered as equal by these parameters, but have different offsets, etc.

Then, the generated fingerprints from two images are compared to define which functions have been changed. It is noted that only fingerprints are compared. Functions are considered "equal" if fingerprints match. If fingerprints do not match, functions are considered as "removed" or "new," depending on whether or not, it is present in the first reference firmware.

Eventually, a similarity threshold is used to perform a more in-depth analysis; two firmware images are compared function by function based on the number of instructions. The number of differences for each instruction is calculated. If this number is bigger than the similarity threshold, then the function is marked as "removed" or "new." If smaller, it is then marked as "similar."

At the end of the process, the comparison algorithm calculates the confidence rate which shows whether numerous changes were made. This rate can be used like a useful but not final reference to establish if gross potential vulnerabilities can affect the new release of the firmware and if it needs to be further analyzed to verify other critical issues.

➢ **Vulnerabilities check by features**

The security vulnerabilities recognition (that can be exploited by potential attacker) present in IoT firmware is implemented by using 3 different features:

- ***Format String Vulnerability Detection***

Based on the signatures, it is made by looking for all the printf functions in the firmware code and analyzes the first parameter that is passed to it. If this is a formatting string like "%s" or "%d", then the code is safe. If we see a memory address or a parameter of the function that has initially called the printf function, then we have a potentially vulnerable place.

- ***Arbitrary Memory Access Detection***

This feature provides functionality for looking through code and detecting all the read and write memory accesses. After such places are detected, analysis is performed to detect whether the address value was changed using some relative value, i.e., whether it has increased or decreased comparing to the base address. Next the obtained offset values are analyzed, and it is determined whether it is allowed to go beyond the boundaries of the allocated memory. Due to the peculiarities of static analysis, values of the parameters from which the functions are called cannot be given and cannot return an exact answer whether this place is vulnerable or not. This leads to false positives. If a metadata provided by the offline static analysis provides a symbolic offset, then the security engine checks the formal reasoning that ensures this metadata and it then uses this metadata to remove the alarm. This can optionally be turned off via analysis options.

- ***Buffer Overflow Detection***

This feature uses signatures to look for the vulnerable functions (strcpy, strcat, memcpy, etc.). After such functions are detected, all the places where these functions are called are determined. Next, the parameters that are passed to them are analyzed to determine if an overflow is possible in this place or not.

11.3 Conclusions and Future Work

As a conclusion, the verification of Security properties in CHARIOT is taken at three levels: offline by the developer, offline by the security engineer, and online and automatically by the CHARIOT Security Engine. Each level does its own verification. An interesting added-value is that the verification done at some level can help the next level to obtain more conclusive results. For example if the Security Engine rejects a firmware because it has doubts about a portion of code, the CHARIOT

dashboard can alert the Security Engineer. Then, the Security Engineer can inspect the code again and add in the meta-data formal explanations about why the code is correct. The next firmware upload should be accepted because the CHARIOT Security Engine will now check the formal reasoning of the Security Engineer in relation with the incriminated portion of code.

Acknowledgments

This project has received funding from the European Union's Horizon 2020 research and innovation program (No. 780075). The authors acknowledge the research outcomes of this publication belonging to the CHARIOT consortium.

References

[1] Proof Carrying Code: https://en.wikipedia.org/wiki/Proof-carrying_code
[2] Github persistent monitor (for static source code analysis, GCC based): https://github.com/bstarynk/bismon
[3] Frama-C Software Analyzers: https://frama-c.com/
[4] Formal Verification for an Internet of Secured Things Tutorial at ACM SIGAPP SAC 2019: https://frama-c.com/download/publications/tutorial_frama-c_for_iot.pdf

DOI: 10.1561/9781680836837.ch12

Chapter 12

End-to-End Security for IoT

By Paul-Emmanuel Brun and Guillemette Massot

According to market forecasts, Internet of Things (IoT) products and services are spreading very quickly in all professional and mass-market usage scenarios in terms of revenues and volumes of devices and services. In the meantime, lots of pilots and commercial deployments are demonstrating the added value of IoT-based solutions in real scale, and in all market segments, from the transportation industry to the utility sector but also in smart building, factory of the future, or healthcare applications. The use of historical and live data in big data lakes will allow recognizing anomalies, implementing alert functions, calculating solutions and optimizations for a more efficient digitalized world. In order to build trustable services and solutions, devices information needs to be trusted. This can only be ensured if IoT systems are end-to-end protected, ensuring protection against corrupted third parties. Indeed, while state-of-the-art IT communication protocols such as HTTPS ensure end-to-end security, many IoT communication protocols have strong constraints in terms of bandwidth, power, and computation capabilities that do not fit state-of-the-art security protocols. In this context, BRAIN-IoT project has tested an innovative approach to ensure end-to-end security on constrained protocols with limited impact on bandwidth and power consumption.

12.1 Introduction

While in the past we had hardware and software in charge of monitoring devices and/or processes in a separate Operational Technology (OT) network, today's IoT systems are directly connected to IT networks that fuel data lakes with historical and real-time data. This in return allows the recognition of anomalies, the implementation of alerts, the finding of solutions, and the dynamic optimization of the overall resources allocation. As modern networks are now formed by hundreds or thousands of connected nodes, the threat of cybersecurity breaches has increased too. To reduce this threats coming from small sensors connected to the cloud and then to the entire network, the IoT devices need to be protected.

Researchers have pointed out security vulnerabilities of Low Power Wide Area Network (LPWAN). Applying traditional IT technologies on Cyber-Physical Systems (CPS) is not a solution as it creates many single points of failure that could lead to data flow disruption and exposure. Another challenge is the use of corrupted edge equipment, which could lead to secret leakage and device identity spoofing.

Moreover the huge number of devices and their specificities lead to complex management activities with high operating costs, especially regarding device identity and security management. Last, LPWAN devices often make trade-offs between cybersecurity and calculation and battery power to reach the promised battery life time, which is typically around 10 years. Indeed, the technological advances in network architectures, operational technologies, Cyber-Physical Systems, and maturation of cloud services have opened new business opportunities for critical and privacy sensitive application. However, the lack of software-based security embedded in the sensors and the proprietary provisioning methods delivered by network providers are preventing the commercial deployment from booming.

In this chapter, we will first have an overview of IoT systems architecture and associated challenges, especially regarding LPWAN technologies introduction in existing architectures.

After the architecture and security challenges overview, we will present relevant associated countermeasures that are available at the state of the art.

Finally, we will describe an innovative approach, allowing to secure very low power devices, which are currently the weakest point to infiltrate an IoT system.

12.2 IoT Systems Architecture and Security Challenges

The search of cost-effective networks covering wide geographical areas combined with the sunset of 2G and the late arrival of the new NarrowBand-IoT (NB-IoT) networks have prompted the emergence of Low Power Wide Area Networks.

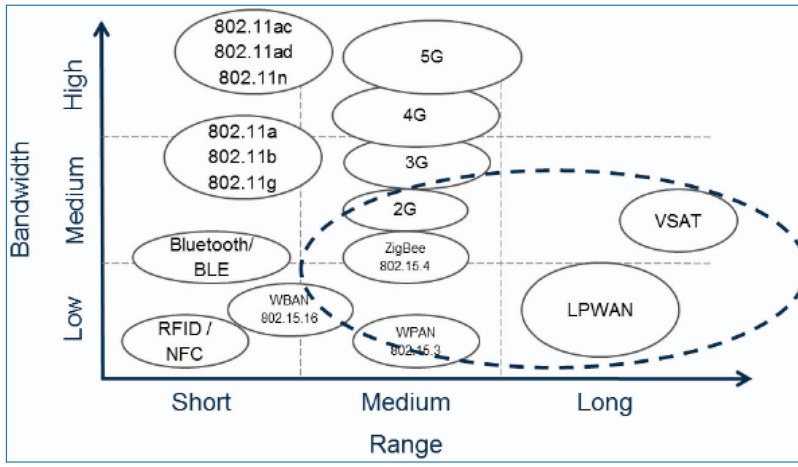


Figure 12.1. Bandwidth versus range capability.

# of LPWAN device shipments in mio	2017	2018	2019	2020	2021	2022	2023
LoRa	17,1	26,0	33,9	42,4	53,2	68,3	89,1
Sigfox	1,2	3,3	7,2	13,0	22,5	38,2	64,6
802.15.4 WAN	16,2	19,3	22,7	27,0	32,1	38,7	46,6
Total	34,5	48,6	63,8	82,4	107,8	145,1	200,4

Figure 12.2. LPWAN & 802.15.4 devices deployments.

LPWAN are a low cost and a reliable alternative to link low battery consumption sensors with long-range (>2 km) as shown in Figure 12.1. At the moment, LoRaWAN and SigFox are the well-known LPWA Networks. However, less known LPWA Networks also exist, such as RPMA, Symphony Link and Weightless.

In 2017, LoRaWAN claimed to have 52 networks publically announced, 350+ ongoing trails and city deployments and presence in over 100 countries worldwide. For its part, SigFox has presence in 36 countries and a coverage of more than 85% of the population in 17 countries. The latest forecasts report that the accumulated number of LPWAN devices shipment between 2018 and 2023 will be approximately 600 Millions as it is highlighted by Figure 12.2.

Those new architectures based on LPWAN communications, combined with OT and Cyber-Physical Systems, open a wide range of new opportunities for business critical and privacy sensitive applications. An exemple of a multi-networks IoT system is presented in Figure 12.3 IoT Industrial Cloud systems are rapidly maturing as packaged systems that can sense, collect, analyze, and action on edge devices. However, the software-based device security and manual/proprietary provisioning methods delivered by platform providers are preventing deployments from scaling.

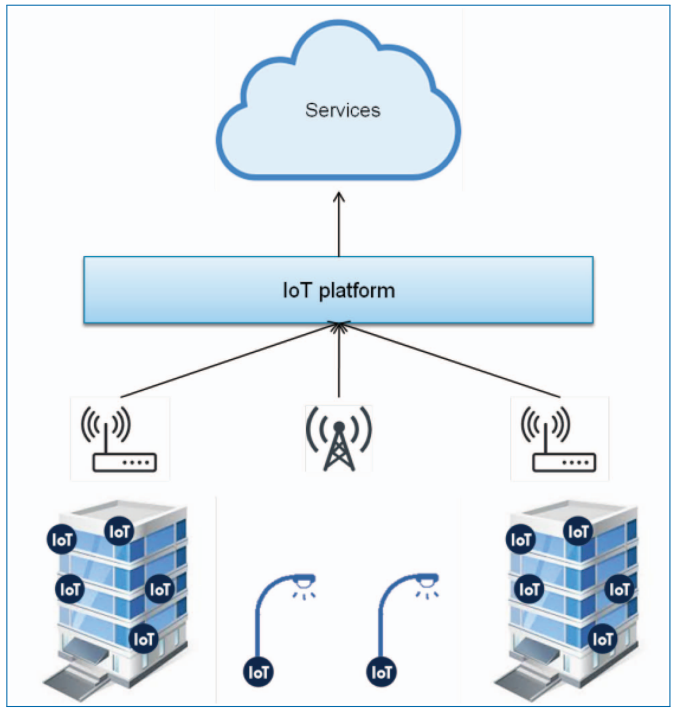


Figure 12.3. Multi-networks IoT systems.

In order to turn those opportunities into reality, some issues need to be addressed:

- **End-to-end security:** Confidence in data is key for services based on IoT. In order to have a "robust" & "trustable" system, data need to be secured, from the IoT sensor to the cloud (or server)-based service applications, in terms of confidentiality, integrity, and availability.
- **Heterogeneity of devices:** LPWAN use the potential of very low power and low-cost CPU, with limited bandwidth and memory but long battery lifetime. Legacy IT protocols do not suit those devices constraints.
- **System integrity as a whole:** The integrity of the system is a key factor to ensure the security and safety of CPS systems. Bootstrapping process and network latency are key issues: devices identities which feed the system services with their data need to be guaranteed and managed.

LPWAN and mesh short-range radio networks today sacrifice security requirements in order (i) not to overload the limited memory and CPU power of very light micro-controllers and (ii) to save energy so that they can reach the promised battery lifetime. Many researchers have pointed out security vulnerabilities of those

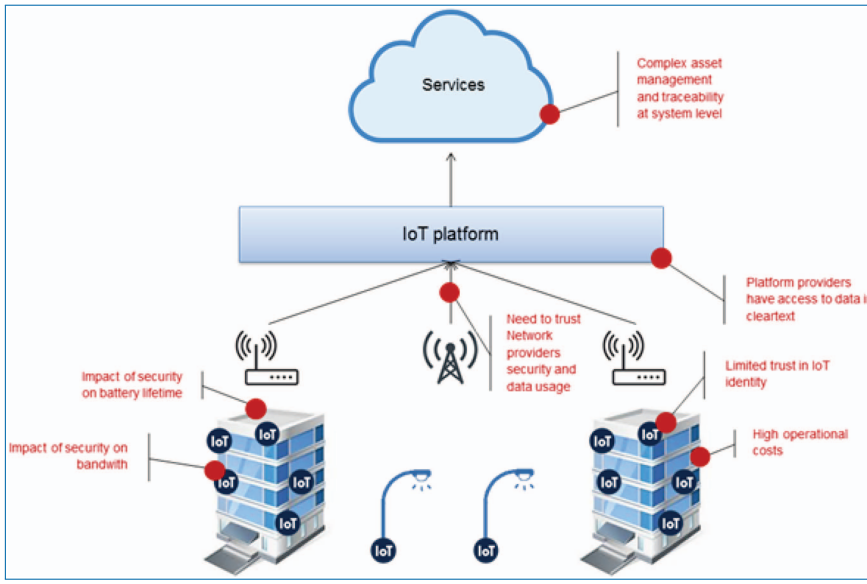

Services
Complex asset management and traceability at system level
IoT platform
Platform providers have access to data in cleartext
Need to trust Network providers security and data usage
Impact of security on battery lifetime
Limited trust in IoT identity
Impact of security on bandwith
High operational costs
IoT

Figure 12.4. Pain points in IoT.

networks. Applying traditional IT technologies on CPS is not a solution as IoT architecture inspired from IT architecture creates many single points of failure. It leads to flow disruption (from the identity and secure channel point of view) and exposes data, device, and application to many threats, such as intermediary equipment corruption, application secret leakage, device identity spoofing, etc …. Moreover, the huge number of devices and their specificities lead to complex management activities with high operating costs, especially regarding device identity management and security management.

In addition to those cyber-security limits, the Figure 12.4 above gives an overview of some technical and financial pain points that need to be addressed to unlock the full potential of low-power and battery-driven IoT.

12.3 State-of-the-Art Mitigation Measures

In order to address the security threats highlighted in the previous section, industrials set up mitigation measures to limit the impact of cyberattacks on those systems. In this section, we will classify mitigation measures in 3 main categories:

- Device management
- Endpoint protection
- Communication security

12.3.1 Device Management

As mentioned in Section 12.2, devices management, and especially security management (identity, cryptographic keys and right management), is very complex within IoT systems, thus leading to high operating costs. In order to reduce those costs and optimize management operations, IoT-related companies such as SigFox, Device Authority, MOCANA, … offer their own solutions.

However, those solutions are limited to the state-of-the-art technologies in place and do not provide a two-factor authentication according to which one unique element (ID or encryption key) is provided by the sensor itself and another unique element is flashed to the sensor from an external library. Solutions offered today for low-power devices are still very complex (no standard for bootstrapping), can fairly easily be attacked, and cause high manual operation cost due to deployment, registration, and maintenance of devices spread over a large geographical area.

12.3.2 Endpoint Protection

In order to enhance the protection of end devices, especially to secure the cryptographic keys, many providers offer secure elements or Trusted Platform Module (TPM) to be put aside the main Micro Controller Unit (MCU). While this approach offers a good solution in terms of security, it also has 3 main drawbacks:

- Updating of keys is **more complex**
- Secure Elements, as dedicated hardware components, **increase the final cost** of the device or could be complex to integrate if not anticipated on the PCB
- There are low-power MCUs out there which have hardware cryptographic optimization – to put extra TPMs would not add value and will burden their lifetime.

In the area of LPWAN, end devices have many constraints such as hardware cost, lifetime duration, bandwidth, and computation power that are not compatible with the use of a dedicated hardware component.

12.3.3 Communication Security

State-of-the-art communication security solutions are mainly based on Transport Layer Security (TLS) or Datagram Transport Layer Security (DTLS) protocols. Combined with the use of certificates Public Key Infrastructure (PKI), those protocols perfectly fit to the constraints of web-based communication with high bandwidth and high power computation. Indeed, the use of certificates

(and asymmetric cryptography) limits the need to share secrets to establish end-to-end secure channel.

The arrival of LPWAN, however, reveals the limits of those technologies:

- Low-power devices **don't offer enough computation power** to implement asymmetric cryptography or certificate management. As an example, the key negotiation requires more than 600 ms of intensive MCU computation using a Cortex M0/M0+, and the full establishment of the TLS session takes 3.2 seconds (using ECDHE for key negotiation protocol).
- LPWAN protocols have a **limited bandwidth**. As an example, the establishment of a secure channel based on TLS uses around 6000 b, while the maximum size of a packet on LoRaWAN network is limited to 250 b.

To overcome those limitations, LPWAN providers added a dedicated security layer in order to protect the transmitted data. Therefore, such solutions result in data security being closely linked to the network protocol. As a result, no end-to-end protection can be applied to the data in cross-technologies networks. In the Figure 12.5 below, any successful cyberattack on the network provider infrastructure or IoT platform concentrator HUB will lead to a leak or a corruption of cleartext data.

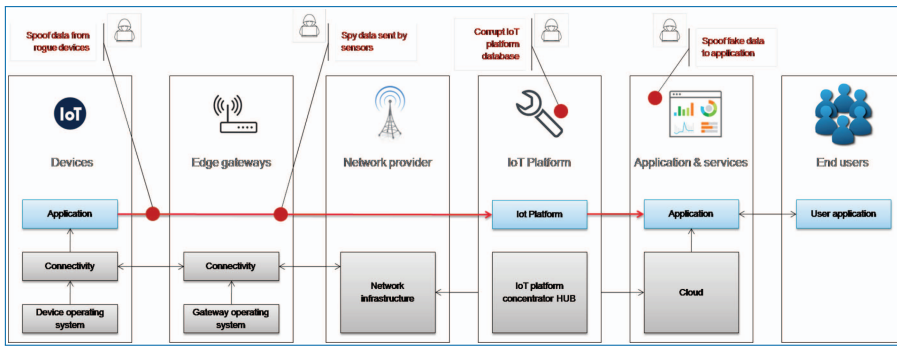


Figure 12.5. IoT communication threats.

Authentication encryption solution	Energy Consumption	Security Level	Management Cost	End-to-end protection
PKI combined with DTLS protocol	⚡⚡⚡	🔒🔒🔒	$ $ $	Yes
LPWAN with IoT Platform management	⚡	🔒	$	No
Secure Element (+ TPM)	⚡⚡	🔒🔒🔒	$ $ $	No
LPWAN without IoT Platform management	⚡	🔒	$ $ $	No

Figure 12.6. Mitigation overview.

The Figure 12.6 above provides an overview of each IoT management solution. We can see that the only solution offering end-to-end protection is highly power consuming and therefore cannot be used to secure low power devices.

12.4 Bring End-to-End Security Layer to Low-power IoT Devices

To overcome the current state-of-the-art limitation, Airbus, through BRAIN-IoT project, is proposing an end-to-end security layer for low power devices, which is independent from the network providers. It brings 3 security features under patent application:

1. **Strong authentication** of devices
2. **End-to-end encryption** of data
3. **Optimized key management**

12.4.1 Strong Authentication of Devices

As a root of trust, any device needs to be secured and authenticated within the system; even the simplest sensors could be the open door to the whole system.

Thanks to Airbus technology, smallest devices, such as cortex M0/M0+ sensors, will be able to encrypt and authenticate any data sent over the network in a stateless mode, dividing by 10 the time needed to establish a secure channel.

The security layer allows aggregation of the device identity and characteristics (such as hardware serial numbers, memory and MCU footprint) into a unique digital ID, which is then combined with a unique cryptographic element for each device to authenticate the data sent by the device and therefore applies the two-factor authentication principle illustrated in Figure 12.7. This in return reinforces the data integrity to build trustable digital services on top of it. To optimize the

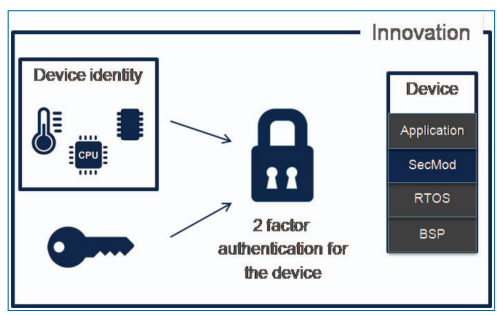


Figure 12.7. Two-factor authentication.

security footprint and to guarantee its robustness, the security layer relies on well-proven authenticated encryption algorithms such as AES-CCM* and AES-GCM and tends to integrate lightweight security protocols such as SPECK to go a step further in energy consumption impact.

Regarding the device authentication, this technology offers the following state-of-the-art advantages:

- For the IoT manufacturer, to add trust at the device layer at reasonable cost by providing a "super-slim" security software
- For the IoT operator, to simplify IoT device deployment with self-registration capabilities and thus to reduce the time and cost necessary to deploy a secure IoT system.

12.4.2 End-to-End Encryption Data

In previous sections, it has been highlighted that edge equipment can have access to the data flow. This potential weak point of the architecture is due to the limits of existing end-to-end security protocols. To prevent this, Airbus security library on edge devices will secure any data sent over the network at application level using the authenticated encryption, as illustrated in Figure 12.8. Therefore, intermediary equipment which use a different protocol will no longer have access to the cleartext data, if not requested by the use case. If requested by the use case, Airbus technology can be applied on edge to cloud authentication as well.

Regarding the end-to-end data security, this technology offers:

- Simple establishment of **need-to-know** principles for any intermediary equipment.

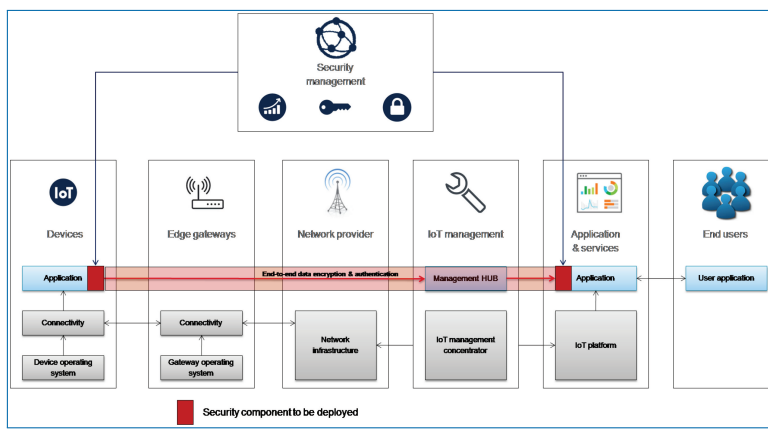


Figure 12.8. Security components.

- **Simple end-to-end security layer** by providing a security library for devices and a gateway to be transparently deployed in front of applications.
- If necessary, **security function** able to decrypt and check data for **edge computing**.

12.4.3 Optimized Key Management

IoT systems are complex to manage because of their "cyber physical" nature, and the huge number of devices and nodes involved. To enable the power of IoT, operating costs need to be reduced.

In Section 12.2 and 12.3.1, we have described the IoT system security management issues. From the security perspective, cryptographic keys and device identity are critical for the trust of the whole system. Therefore, managing **millions of secret keys** in centralized database on IoT platform and application could lead to expensive security measures to guarantee their security. Using Airbus security manager to generate and manage the key lifecycle will allow applications and IoT platforms to **compute each device key on the fly** based on one master key which could be secured using a legacy vault.

Our innovative key management method is fully based on key hierarchy technologies using well-proven derivation algorithms such as HKDF (HMAC-based Key Derivation Function) and is illustrated in Figure 12.9.

This enhances the key management system in order to support symmetric cryptography and therefore, according to Airbus tests, allows to gain about 50% of battery lifetime in regard to PKI performance. A comparison of Airbus solution capabilities compared to other existing solutions is described in Figure 12.10.

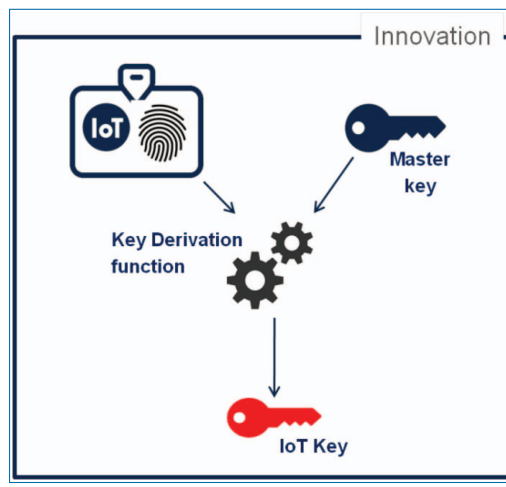


Figure 12.9. Key derivation approach.

Authentication encryption solution	Hardware cost	Time overhead for the security layer	Energy Consumption	Security Level	Management Cost	End-to-end protection
PKI combined with DTLS protocol	$$$	600 ms	⚡⚡⚡	🔒🔒🔒	$$$	Yes
LPWAN with IoT Platform management	$	50 ms	⚡	🔒	$	No
Secure Element (+ TPM)	$$$	35 ms	⚡⚡	🔒🔒🔒	$$$	No
LPWAN without IoT Platform management	$	47 ms	⚡	🔒	$$$	No
Our solution	$	50 ms	⚡	🔒🔒🔒	$	Yes

Figure 12.10. Solution overview.

Moreover, the security manager will store identity information combined with associated privileges, allowing fine grain control of edge device access to applications. Such security functions also limit propagation of cyberattacks on the IoT systems by suspending suspicious devices from this security management service.

This solution enhances device security management and operation, thanks to the security management service allowing:

- **Accelerated bootstrapping** with simplified key and identity provisioning through Airbus key generation technology based on master key derivation
- **Fine grain access control** from devices to application
- **Fast reaction to cyberattack** through interconnection with the security operation center.

To conclude, this solution combines well-proven algorithms with an innovative approach in order to solve the issue of end-to-end security for large number of constrained devices, spread into constrained radio protocols such as LPWAN or 802.15.4.

12.5 Conclusion

This security solution unlocks the potential of LPWAN applications in Machine to Machine (M2M), device to edge, edge to cloud, and device to cloud communication by providing end-to-end security for IoT systems independently of networks

including slim encryption layer to ensure trustable and optimized data transmission. This solution also allows self-registration of devices in a plug & play mode, ensuring automatic key renewal over constrained protocols.

This solution combines perfectly with existing security measures such as use of secure element and network security capabilities to ensure the trustability and the confidentiality of data, which is key to build new digital services.

Acknoweledgments

The work presented in this Chapter has received funding from the European Union Horizon 2020 research and innovation programme under grant agreement No 780089, Brain-IoT project.

DOI: 10.1561/9781680836837.ch13

Chapter 13

Blockchain Ledger Solution Affirming Physical, Operational, and Functional Changes in an IoT System

By Alexandros Papageorgiou, Konstantinos Loupos and Thomas Krousarlis

CHARIOT aims to introduce a novel methodology for maintaining security throughout all endpoints of an IoT network. To achieve this, a blockchain-based adaptation of the traditional Public Key Infrastructure (PKI) protocols have been formulated to satisfy the needs of industrial IoT and enhance it as necessary to simultaneously be as abstractive as possible in its design and fulfill the requirements set forth by the enactors of the CHARIOT technology stack. The current technological landscape of identity security in the context of IoT lacks in several areas, even more so when analyzed under the pretense of low-resource devices. The most common solution applied in these scenarios is the use and deployment of a PKI. However, it fails to address the core issues of identity security in high-value low-resource distributed environments since PKI has its shortcomings in certain security aspects by design.

The first and foremost shortcoming is that of the single point of failure [1], a trait shared among any software design implementation that relies on a one-to-many server-to-client interaction. By introducing a single source of truth for cross-verification of identities, it enables an attacker to devote his resources into compromising that single source instead of relying on other more complex attack strategies. Additionally, should a network-wide disruption occur, it is relatively

easier to spoof a single entity rather than a distributed system and validate malicious identities. As more trust and value, informational or otherwise, is placed in IoT networks, these types of attacks will become much more widespread as the beneficial yield would outweigh the resources required to take such a network down.

To address this, the abstraction of the PKI stack is proposed to a distributed layout with the aid of blockchain technology [2]. In PKIs, Certificate Authorities (CA) act as the validation oracles for any identities in the network by issuing identity certificates and subsequently validating them cryptographically. These authorities possess a single cryptographic keypair that, if compromised, can result in devastating consequences as the possessor of the keypair would be able to issue, adapt, and revoke an unlimited number of identities, thus resulting in the compromisation of the whole network.

CHARIOT decided to break down the components that make up the issuance and verification process of a CA, simplifying them where possible and finally reimplement them using blockchain-based smart contracts. As an initial step, the bare necessities of a PKI network were defined in association with a cryptographic identity, thus an identifier, with a set of metadata that could either reside in on-chain data or simply act as a validator of off-chain data, such as a cryptographically secure hash. To enhance the basic indications of a PKI identity, which are "valid," "expired," or "revoked," an additional field is added to signal the status of the identity. This field takes the form of an unsigned integer, which is utilized in CHARIOT solution as a binary abelian group code. This code can be used to represent an arbitrary amount of statuses and, being a simple representation of ones and zeroes, depends on the software implementation for assigning a binary group to a status code. This results in a modular system that can easily be expanded in scope.

By using smart contracts, we ensure that all on-chain computations will be relayed across the network and ensure that any alterations to the underlying database will be available to queriors of the blockchain instance from any blockchain node on the network. This allows the network to remain operational even if a percentage of the network is compromised. Although in the CHARIOT implementation, the IoT gateways were utilized as nodes in the blockchain network, as the forthcoming technological advancement in the IoT's computational capabilities will enable them to become blockchain nodes themselves, empowering verification of identities in a bilateral format (both gateway and sensor identities verified) rather than a unilateral format (only sensor identities verified).

With regard to the actual invocation process of a contract's identity issuance, revocation, and adaptation method, a multi-signature protocol has been implemented rather than a single-signature one. As the acceptance policy is preconfigured during the setup of the CHARIOT solution, it is possible to construct an n-out-of-m scheme that would satisfy even the most rigorous security policies.

For instance, for the installation of a sensor, the four-eye principle could be applied by requiring signatures by both the installer of the sensor and the manager of the site in which the sensor is installed.

13.1 CHARIOT Blockchain Requirements

In parallel, a custom-built cryptographic protocol was utilized rather than a readily made one as that allowed greater freedom in the implementation approach as well to ensure that it would be able to be adapted if it is necessary in the future. To construct cryptographic signatures that would be accepted by the CHARIOT blockchain system, one needs to sign a payload that matches the CHARIOT blockchain requirements. These requirements are as follows:

(a) All members of the payload utilize the hyphen character "-" with ASCII code 45.
(b) The first member of the payload is a shorthand for the type of operation. These shorthands are defined in the smart contract directly and represent actions on the types of assets stored on the blockchain. Here, assets refer to the types of identities handled in the blockchain. For the purposes of CHARIOT, a set of assets have been defined that are applicable in this case: "Software", "Sensor," "Service," and "Administrator." These assets have been assigned the following shorthands: "sw," "ss," "se," and "ad," respectively. The types of assets and their corresponding shorthands are entirely implementation dependent and are in no way restraint by the underlying implementation of CHARIOT. By following the conventional Create-Read-Update-Delete operations, commonly referred to as CRUD, the types of operations are defined as the initial of the operation followed by the shorthand of the asset type, e.g., css means (c)reate (s)en(s)or, whereas usw means (u)pdate (s)oft(w)are.
(c) The second member of the payload is the nonce of the identity signing the payload. If we simply sign a static payload that would not differentiate between subsequent signatures of the same action on the same asset, an attacker would be able to intercept a signature and replay it across the network to, for example, re-instantiate a previously decommissioned sensor. To prevent this, the same nonce-based system has been adopted in use by the Ethereum blockchain [3], a linearly incremental unsigned integer that begins at zero and increases by one after each transaction accepted in the network. This nonce is directly stored on the blockchain itself and is retrievable by all nodes to enable proper validation of incoming signatures at all endpoints.

(d) The third and final member of the payload is the identifier of the asset we wish to alter. This identifier usually takes the form of the encoded public key of the identity, although the protocol we have devised supports an arbitrary identifier.

The aforementioned specs lead to the creation of a pseudo-language, also known as a Domain-specific Language (DSL), that is machine-readable and easily convertible to a human-readable format. Additionally, the nonce-based scheme heightens the security of the overall solution as it provides protection against a wide variety of attacks, inclusive of replay attacks as well as race conditions, two issues that exist inherently in any cryptographic signature-based scheme.

In order to ensure that the specifications of the blockchain platform comply with the requirements of the testbeds the CHARIOT project has already been deployed to, feedback was gathered during the blockchain's design from the owners of the testbeds and incorporated them to the CHARIOT blockchain when necessary in collaboration with the other components of CHARIOT. One such requirement was the increase of security at the services themselves, as the initial vision was for the CHARIOT services to run in a non-distributed server device that still suffered from certain types of vulnerabilities. The new requirement of distributed deployment was tackled by introducing the services as identities within the distributed PKI network formed by the blockchain and subjects them to the same stringent security workflows identities on the blockchain are put through. In turn, this enabled greater control and security of the services running in the CHARIOT network. As an additional enhancement, the presence of metadata was introduced within the identities in the blockchain network to indicate the cryptographic keys that would be used to encrypt data payloads in an end-to-end fashion to accommodate for the privacy requirements of the data, as is the case when the Privacy engine of CHARIOT processes data flows in the network.

13.2 CHARIOT Technology Stack

13.2.1 Administrator Keypair Manager

To streamline interoperability integrations of the blockchain service of CHARIOT, several tools and services were developed and are included in the CHARIOT technology stack, the first being the Golang CLI CHARIOT administrator keypair manager. As the scheme that has mentioned above might be slightly complex to implement, prone to human error and difficult to carry out in a manual way, a tool was developed that allows simple command-line arguments to be parsed and generate keypairs and signatures with. Additionally, it is capable of both outputting and consuming CHARIOT keypairs as exported to the .chrt file format, a file format

devised in CHARIOT that is a purely textual document structured similarly to conventional keypair files, as in the keypairs are enclosed within multiple dash delimiters with a word in between. Apart from the thorough documentation provided alongside the tool, it was also used to directly implement an externally queryable JSON-RPC 2.0 API to allow ease of integration with existing tools of the CHARIOT technology stack that reside within the main CHARIOT network. As the tool itself is stateless, it is possible to easily scale it out as necessary by spawning more processes running the same software.

13.2.2 RESTful API

Another tool that was developed for the purposes of integration with the blockchain of CHARIOT is the RESTful API coded for the Node.JS runtime environment. Interfacing with the Hyperledger Fabric blockchain instance is notoriously difficult to non-blockchain-versed developers, and as such, it is naïve to expect both users and other members of the CHARIOT consortium to integrate the blockchain easily without additional assistance [4]. To amend this, the Fastify framework was used along with Hyperledger Fabric's own Node.JS libraries to create a bridge between the externally facing API and the internally facing blockchain interface. The Fastify server essentially transforms the RESTful requests it receives to actionable items on the blockchain instance and returns the results where applicable. While the blockchain itself contains an additional layer of traditional PKI, this was abstracted by not deploying a CA for the blockchain network and instead generating and utilizing a singleton administrator keypair when interacting with the blockchain via the API. While one may argue the storage and utilization of a single keypair for interacting with the blockchain may be vulnerable to security issues, this is not the case with the CHARIOT blockchain implementation as the validity of the transactions themselves is processed directly on the smart contract without any reliance on the blockchain's native PKI system. This means that, even if a malicious transaction was crafted with the keypair, it would immediately be rejected by the smart contract as its code is immutable. Even if an attacker attempts to alter the codebase and create a new version of the contract, which would be possible if he had possession of the administrator keypair, he/she would still fail to compromise the network as Hyperledger Fabric's versioning system prevents the replacement of a contract's chaincode with another's.

13.2.3 New Genesis Transactions

Another limitation of the Hyperledger Fabric blockchain implementation was the non-dynamic structure of the network. By default, version 1.0.6 of the Hyperledger Fabric runtime lacks the dynamic network restructuring subsequent

versions inherently support. To emulate this capability, CHARIOT took advantage of some of the configuration tools of the blockchain to set new genesis transactions that would replace the initial genesis transaction of the network and thus allow dynamic addition and removal of Orderers, the nodes responsible for maintaining the blockchain network's state synchronous and without any reconciliation issues. This case arises when the CHARIOT network needs to scale out by adding numerous more IoT gateways and generic network participants, as maintaining a single Orderer for an extensive network would prove cumbersome and ultimately unsupportable.

13.2.4 Cryptographic Implementation

The underlying cryptographic implementation used for the CHARIOT keypairs and general signature protocol is based on Elliptic Curve cryptography, specifically the ecdsa package of the Golang programming language which the smart contract codebase is programmed in. An exhaustive analysis of the available state-of-the-art cryptographic suites was conducted at the time of CHARIOT's initiation period and pinpointed that the ideal suite for the purposes was, at the time, the ecdsa package. However, this is not an inhibitor of the future relevance of the CHARIOT blockchain solution as the implementation itself is suite-agnostic provided that the relevant components are properly updated and/or altered to utilize a different cryptographic suite. For example, one could introduce a quantum-secure digital signature algorithm in the future to ensure that the blockchain solution remains secure even when put under attack by quantum computers. This extensibility capability expands to all other aspects of the implementation. The CHARIOT implementation uses certain technology stacks to carry out its vision of a distributed PKI system, namely Hyperledger Fabric, a Node.JS-based RESTful API, and a Golang CLI tool. These software restrictions are irrelevant to the logic of the overall blockchain workflow, as it can be ported to any programming language one wishes. This type of abstraction is detailed within the project and one can use a completely different blockchain implementation, e.g., Ethereum, to achieve the same results as one would with Hyperledger Fabric.

The rationale behind the software-dependent choices were made based on research conducted over the lifetime of the CHARIOT project coupled with hardware limitations of the CHARIOT gateway built within the project. As is the case with most industrial grade IoT gateways, the CHARIOT gateway operates in a real-time operating system environment, meaning that it does not pair well with event-driven or asynchronous software. This proved to be a strong obstacle for the blockchain implementation of CHARIOT, since it was aspired to boast a distributed network of blockchain participants inclusive of the IoT gateways. For this purpose, a System-on-Module device was integrated that would

be capable of operating a conventional operating system and create it as an adapter between the real-time environment of the gateway and the asynchronous nature of the blockchain. By forming a wired bridge between the System-on-Module and the motherboard of the CHARIOT gateway using Universally Asynchronous Receivers-Transmitters, were able to relay data gathered from the blockchains' latest state to the gateway whilst also being able to respond to actuations made by the IoT gateway and properly convey the result of these actuations, such as on-demand queries of a sensor's identity.

The System-on-Module itself needed to satisfy the bare minimum operational requirements of a blockchain instance which meant that a powerful yet compact System-on-Module has to be attached to the gateway. Power to it was provided through direct tapping of the gateway's power source. A new issue that arose with using a System-on-Module was the available software implementations of the blockchain in specific architectures. The software runtime of the Hyperledger Fabric project did not include the ARM family of architectures, and as such, CHARIOT had to resort to building its own implementation from the source code. This proved to be a hindrance, as no one had compiled the Hyperledger Fabric runtime for ARMv8, the 64-bit architecture of the System-on-Module. Significant alterations have to be done to the build instructions and choose a specific Hyperledger Fabric version to continue with to ensure full compatibility with the API of the blockchain. In the end, CHARIOT went forward with the 1.0.6 version as the 1.1.0 version, the latest available one at the time, appeared incompatible with the ARM architecture after attempts at compiling it inside the environment, meaning it required heavy alterations that were out of scope of the project.

13.3 Scalability, Integration, and Testing

While the blockchain itself is inherently scalable due to its segregated structure, the overlaying services needed to be somehow adapted to exemplarily operate under a scalability scenario. To this end, deployment scripts were coded to automate the process of spawning an instance of the software and used these scripts to create deployment pipelines that contain instructions on how to deploy the various components of the CHARIOT blockchain in the Docker environment. By storing the code of the components in virtualized containers, overlay services were set up to manage the life cycle of these containers automatically and scale the network as necessary, ensuring that the systems remain operational even under extreme network load. Additionally, load-balancing services can easily be appended on top of the architecture to properly delegate network load to the containers spawned underneath the load balancer.

The environments the CHARIOT project is and will be deployed in are extremely time-sensitive, as even a minor delay in the measurement of a critical sensor can lead to catastrophic results. To properly conduct tests on the blockchain solution and ensure that it does not become the bottleneck of the overall solution, multiple unit tests have been devised that thoroughly check the correct operation of the underlying logic. These unit tests were coded with a framework that simultaneously produced coverage reports, enabling us to identify which segments of the codebase were not thoroughly tested and required extra test suites. While these unit tests provided us with an indicator of the correct operation of the logic, they failed to provide any insight as to how the solution fares under extreme load. For this purpose, load-testing tools produced multiple HTTP requests under a multi-threaded context and tracked the resulting response times, codes, and other such metadata. The results gathered from the tests indicated that the blockchain provides a small yet non-negligible overhead, and as such, optimization techniques opted to speed up the identity validation process of the solution.

One such optimization was the introduction of an in-memory table of recently verified identities that are retrieved and retained on boot time at the gateway level. This method acts similarly to how popular cache-based data stores work such as Redis. The resulting speed was near real time, as the codebase migrated for extracting the public key of a signature payload from the System-on-Module to the gateway's codebase in C thus decreasing the necessary execution time and allowing the gateway to self-sufficiently verify an identity provided that it exists in its cache table.

13.4 Conclusions and Future Work

The blockchain solution of CHARIOT acts as proof of a viable blockchain integration in the context of IIoT security, and it paves the way for wider adoption of this novel technology; if utilized correctly, it can resolve certain issues faced by contemporary solutions. As future steps, we aspire to enhance the solution over the course of the CHARIOT project with the requirements from the other services of the solution stack as well as the requirements of the pilot cases themselves.

Acknowledgments

This project has received funding from the European Union's Horizon 2020 research and innovation program (No. 780075). The authors acknowledge the research outcomes of this publication belonging to the CHARIOT consortium.

References

[1] R. Housely, "Public key infrastructure (PKI)." in The Internet Encyclopedia (2004).

[2] J. Linn, "An examination of asserted PKI issues and proposed alternatives." (2004).

[3] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger." Ethereum project yellow paper 151, no. 2014 (2014): 1–32.

[4] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart *et al.* "Hyperledger fabric: a distributed operating system for permissioned blockchains." In Proceedings of the Thirteenth EuroSys Conference, p. 30. ACM, 2018.

DOI: 10.1561/9781680836837.ch14

Chapter 14

Leveraging Interledger Technologies in IoT Security Risk Management

By Dmitrij Lagutin, Yki Kortesniemi, Vasilios A. Siris, Nikos Fotiou, George C. Polyzos and Lei Wu

There are security vulnerabilities in all technological systems but particularly in many Internet of Things (IoT) solutions. *Responsible disclosure* has been the established approach for the security community to deal with the discovered vulnerabilities, but this approach does not fare well in the modern fast-paced world and, in particular, in the low-cost, often highly constrained, long-expected usable lifetime, yet highly volatile IoT space. This chapter proposes a Distributed Ledger Technology (DLT) and interledger-based *Automated Responsible Disclosure* (ARD) solution that provides stronger incentives to the involved parties to address the vulnerabilities in a timely manner, better accountability of all parties, and, as a result, better security for the public at large.

14.1 Introduction

Responsible disclosure is a model to disclose security vulnerabilities in a way that allows the vendor some time to create a fix before the vulnerability is publicly revealed. Responsible disclosure is best known and associated with software products and bugs in software, but it applies to both software and hardware, and more generally any products with security features. Most security experts use this model to report vulnerabilities [1].

This model is of particular interest in the context of the Internet of Things (IoT) for several reasons. First, there have been many significant security vulnerabilities in IoT devices. Not only are many IoT devices relatively unsophisticated and often very limited in capabilities, but they are also often more exposed yet less frequently or even completely unmonitored during their operation for extended periods of time. Furthermore, because of the relatively low price and margin of IoT devices, and often fast progress in the domain, many manufactures are disinterested in addressing vulnerabilities publicised in their products. Many of the products have only short market lifetimes and all compete for very fast time to market, leading to an abundance of security issues in products. Finally, because of all these problems, IoT devices have been exploited and have caused recently highly publicized security problems with significant impact [2, 3], and these problems are unlikely to significantly diminish in the near future.

The idea behind the responsible disclosure is sound, but unfortunately it often does not work that well in practice. In many situations, there are no clear and enforced time limits after which the vulnerability will be disclosed, and the vendors behind vulnerable products often request extensions to the disclosure if they are not able to provide fixes in time. Also, there have been several cases where the security experts have been pressured, threatened, or even sued by vendors to not release the vulnerability within a reasonable time frame [4, 5]. As a recent example from August 2019, a security researcher found a vulnerability in Valve's Steam gaming client, but as the vendor was not interested in fixing the issue, the researcher then wanted to publish the findings, and the vendor promptly banned the researcher from the HackerOne[1] bug bounty platform [6]. Furthermore, due to the lack of transparency to the details of the vulnerability, the vendor may release a fix that does not properly address the vulnerability merely to appear as security conscious. And finally, it is impossible for the customers and the public in general to know how many vulnerabilities in each vendor's products have been identified and how many of them remain unfixed.

Even if most vendors put emphasis on security in their advertising, in reality vendors often do not have clear incentives to create secure products or fix their vulnerabilities quickly, which is apparent from a simple cost-benefit analysis:

1. Vendors are typically companies that aim to maximize their profits, which can be expressed as: revenue-expenses.
2. It is not possible for a vendor to generate significant extra revenue by improving the security of its products, since proving or measuring security of the product is practically impossible (it cannot be practically proved that a product is secure, only that it is insecure).

1. https://www.hackerone.com/

3. Similarly, insecure products will not make the vendor lose a significant amount of revenue, since the current Responsible Disclosure scheme does not reveal the number of vulnerabilities to the general public and allows the vendor ample time to fix its products. Since most of vulnerabilities are released to the general public only after a long period of time, most of customers do not suffer too much or too often from using insecure products, and therefore, they do not have a clear incentive to switch to another vendor.
4. Creating more secure products is expensive (it requires more development time, more experienced developers, better processes, more time for testing, etc.); the vendor will often rather minimise the costs by making less secure products.

Per 2 and 3 above, vendor revenue will not be adversely affected by insecure products, yet per 4, creating less secure products reduces expenses. Therefore, per 1, providing less secure products tends to maximise the vendor's profits.

Now, if the vulnerabilities would be disclosed in more strict and transparent manner, the situation would change. Vendors behind insecure products would lose reputation and would have a harder time selling products. Customers could, for instance, demand agreements from vendors with bad security history that in the case a serious vulnerability is found in the product, the vendor would pay compensation to the customer, etc. As a result, vendors would have strong incentives to improve the security of their products, and overall, there would be financial incentives to provide more secure products and fix vulnerabilities quickly, which in turn would lead to more secure products in the market.

Transparency of vulnerabilities is also important because even if the vulnerability has been found and responsibly disclosed by a one expert, it does not mean that no one else in the world knows about it. In fact, it is very likely that the same vulnerability has already been found by other actors, such as criminal organisations, which do not have any incentives to disclose it to the vendor and commonly trade vulnerabilities on black markets [7]. From the customers' point of view, it is categorically better to know that a vulnerability exists (even if the whole world knows about it), compared to the situation where the vulnerability exists but the customers do not know about it. At least in the former case the customers can take actions to mitigate the risk (stop using the vulnerable product temporarily, install additional safeguards, etc.), while in the latter case the customers cannot predict or mitigate attacks in any way since they are not aware of the possibility of the attack. Therefore, a long time between the vulnerability discovery and the related public disclosure increases the possibility of security breaches and decreases the overall security.

To succeed, any practical implementation of Responsible Disclosure also has to be able to deal with the scale of vulnerabilities, so because of the huge number of IoT devices with diverse capabilities and operations, a fully—or almost

fully—automated set of procedures for recording vulnerabilities and their corresponding patches in a transparent and non-repudiated way is necessary to ensure the security and ultimately the success of IoT deployments.

Distributed Ledger Technologies (DLTs) and interledger mechanisms can be leveraged to immutably record when a vulnerability is disclosed to authorities and/or vendor, and after a certain time period (which should be the same regardless of the vendor in question) the vulnerability is then automatically revealed to the public (regardless of wherever it has been fixed or not), or at least the metadata would be revealed (that vendor X did not fix the vulnerability within a certain timeframe). On a technical level, this would be implemented by putting the vulnerability disclosure, *V*, to a private ledger for security authorities and vendor or vendors, and recording the hash of the vulnerability along with the vendor, product, and time information on a public ledger. As a result, the vendors would have stronger incentives to improve the security of their products.

This chapter is organised as follows: Section 14.2 provides background on DLTs, smart contracts and chaincode, the corresponding functionality in permissioned blockchains, interledger technologies, and Decentralized Identifiers (DIDs). Section 14.3 then briefly presents the history of and the state of the art in vulnerability disclosure approaches. Section 14.4 gives an overview of the proposed Automated Responsible Disclosure (ARD) approach of IoT risk management through interledger-enabled disclosure with accountability. Section 14.5 provides an evaluation of the design, and Section 14.6 presents the conclusions and proposed future work in order to establish, promote, and practically evaluate the approach in the real world.

14.2 Background

The Automated Responsible Disclosure (ARD) solution proposed in this chapter builds on four key technologies: distributed ledger technologies, smart contracts and chaincode, interledger functionality, and Decentralised Identifiers.

14.2.1 Distributed Ledger Technologies (DLTs)

Distributed Ledger Technologies (DLTs), such as blockchains, offer decentralised solutions for collaboration, interoperability, and trust. One of the main features of DLTs is the *immutability* of data: ledgers are append-only databases where existing data cannot be modified and only new data can be added. Another major feature of DLTs is a distributed *consensus mechanism* [8], which controls what and how data are added to the ledger. Finally, DLTs also *replicate* data to participating nodes thus improving availability. Because of these three properties, DLTs avoid single points

of failure and offer resilience against many attacks. It is relatively easy to determine if any of the participating nodes in the DLT are misbehaving and even in an extreme case, where an attacker manages to control the majority of the DLT's resources, the attacker still cannot modify the existing data, only control the addition of new data.

DLTs can be implemented with different levels of openness. They can be fully *open* (permissionless), which means that anyone can join the DLT and propose transactions; most well-known DLTs such as Bitcoin and Ethereum are based on this principle. However, DLTs can also be permissioned, which makes them either *semi-open*, in which case read access is open to everyone but write access is restricted, or *closed*, in which case both read and write access are restricted.

The main practical innovation of DLTs is the enablement of distributed trust. While there have been multiple proposals for distributed databases in the past, they have mostly concentrated on the distributed implementation, while the trust model has remained firmly centralised. In contrast, DLTs allow various entities that may not fully trust each other, such as individuals, organisations, and companies, to collaborate in a safe and transparent manner, with only a low risk of being cheated by others.

14.2.2 Smart Contracts and Chaincode

Smart contracts [9] are another important feature provided by several DLTs: they are distributed applications that are executed on the ledger. Whenever an entity interacts with a smart contract, these operations are executed by all (full) nodes in the DLT network in a deterministic and reliable way; one of these nodes is then selected to store the contract's execution outcome (if any) in the ledger. Smart contracts can verify DLT identities and digital signatures, perform general purpose computations, and invoke other smart contracts. The code of the smart contract is immutable and cannot be modified even by its owner. Moreover, since all transactions sent to a contract are recorded in the DLT, it is possible to obtain all historical values of the contract. Smart contracts typically refer to code running on the Ethereum (in which case they are Turing-complete), but similar functionality is available in other DLTs. In particular, in the permissioned Hyperledger Fabric, similar functionality is called *chaincode*, and simpler, more constrained scripts can also be run on Bitcoin. Smart contracts or similar functionality is critical for automating processes and will be exploited in the techniques described later.

14.2.3 Interledger Functionality

There exists a large number of DLTs each offering different trade-offs in terms of latency, throughput, consensus algorithm, functionality, etc., thus rendering them suitable for different types of applications. For example, a DLT can focus on

cryptocurrency payments, recording of IoT events, or access authorisation. In complex systems, it is therefore often not feasible of efficient to use only a single DLT for everything; hence, the *interledger* approach that allows different DLTs to exchange data with each other is required in many situations. Using multiple ledgers is also beneficial for privacy reasons: participants within a DLT need to be able to access all data stored in that DLT to independently verify its integrity, which encourages the participants to use private ledgers and store only a subset of the data to the main ledger used for collaboration with others. A concrete example of the use of interledger is the following: data are stored in a closed ledger and simultaneously related information (e.g., a hash of the original data) is stored to a semi-open or open ledger. At a later time, the original data are revealed and the hash in the (more) open ledger guarantees that the original data have not been modified.

Multiple ledgers are also necessary for crypto-agility, as cryptographic algorithms used by DLTs (such as SHA-256) will not stay safe forever; thus, it is necessary to have a mechanism to transfer data from one ledger to another. Siris *et al.* [10] present a review of interledger approaches, which differ in their support for transferring and/or trading value between ledgers, whether they support the transfer of information in addition to payments across ledgers, the balance between decentralized trust and cost (which can include both transaction cost and delay), the level of privacy, and their overall scalability and functionality that can facilitate the innovation of the DLT ecosystem. Finally, interledger functionality may also utilize *hash-locks*, which are cryptographic locks that can be unlocked by revealing a secret whose hash matches with the value configured in the hash-lock. By using the same secret value on two or more ledgers, it is possible to achieve cross-ledger atomic operations, where the first transaction reveals the secret, which in turn is used to unlock the transaction on the other ledger(s). If the secret is never revealed, none of the transactions will succeed.

Interledger mechanisms, which in the above example involve the hash of data stored in a private ledger, allow securely linking information and/or transactions stored on multiple ledgers. Such mechanisms can be used to create a dependence relation that allows the execution of smart contracts functions on one ledger only if some data has been recorded or transactions have been performed on another ledger. As discussed in Section 14.4 that describes the ARD approach, the above feature ensures that the time dependency and order of events on the public and private ledgers are automatically and transparently enforced.

14.2.4 Decentralized Identifiers (DIDs)

Currently, an identity technology receiving much attention is the *Decentralized Identifiers (DIDs)*. A key aspect of DIDs is that they are designed not to

be dependent on a central issuing party (Identity Provider or IdP) that creates and controls the identity. Instead, DIDs are managed by the identity owner (or a guardian on the owner's behalf), an approach known as *self-sovereign identity* [11].

There are several different DID technologies in development [12], some of the most prominent being Sovrin, uPort, and Veres One. These technologies started with similar but individual goals in mind, but lately many of them have adopted the approach and format of the W3C DID specification [13] thus rendering them more and more interoperable. The specification defines a DID as a random string, often derived from the public key used with the identity. If a new DID is allocated for every party one operates/communicates with, correlating one's activities with different parties would be significantly harder to achieve. This property can be further enhanced by replacing existing DIDs with new ones at suitable intervals, even after just a single use.

Yet, DIDs alone do not suffice, as some means of distributing the related public keys, any later changes to the keys, or other identity-related information is required. To this end, many of the DID solutions rely on a DLT for public DIDs (used by parties that want to be known publicly), whereas for private DIDs (e.g., used by individuals or their personal IoT devices) an application specific channel is used to distribute the information. Some DID technologies, e.g., Sovrin and Veres One, are launching their own permissioned DLTs, while others rely on existing blockchains (e.g., uPort is built on top of Ethereum). All three example technologies originally intended to use DLTs/blockchains for distributing information about private DIDs, but the emergence of the General Data Protection Regulation (GDPR) [14] in the EU and other similar changes have made storing personally identifiable information on a non-mutable platform such as a DLT/blockchain problematic. For this reason, Sovrin and Veres One have already excluded individuals DIDs from the ledger, and similar treatment may face the DIDs of IoT devices if they reveal personal information about their owner.

In ARD, DIDs are used to protect the privacy of security experts discovering the vulnerabilities to prevent any undue pressure to suppress the findings.

14.3 Previous Work

Disclosure of vulnerabilities has long been an important challenge in software industry, because security vulnerabilities are among the major reasons for security breaches [15], and vulnerabilities have been extensively exploited in many successful attacks [16]. Since IoT devices interact with the real world, there are potentially high risks with severe impacts (in particular with actuation directly leading

to safety concerns). At the same time, the growing number of IoT devices introduced in recent years and the new types of data coming with them have increased the security risks both for the industry [17] and the in consumer IoT domain [18]. As an example, a case study on baby monitors [19] details how many baby monitors ship with static credentials and other vulnerabilities allowing access to the confidential information provided by the devices. Together, the above-mentioned risks in software systems and the flaws in the IoT systems can be highly dangerous and cause significant economic losses [20], which draws considerable discussion on how such risks can be controlled and how related vulnerabilities should be disclosed [21].

Over time, many different approaches have been proposed for the mechanisms of vulnerabilities disclosure. Traditionally, the so-called *full vendor disclosure* has been used; there the vendor has full control over how and when a vulnerability is disclosed, which leaves many vulnerabilities totally unfixed or fixed only after a long delay [22, 23]. In order to solve the problem of full vendor disclosure with public at risk, the *immediate public disclosure* has been proposed, which introduces a direct and strong incentive for the related vendor to fix the released issue as fast as possible [24]. Even though early disclosure of the vulnerabilities can drive faster fixing and uptake of mitigating measures by related parties to reduce the risk, the attackers can also exploit the released vulnerability to attack already before a patch is delivered by the vendor [25]. To address the conflicts between vendor and users, a hybrid disclosure approach, *responsible disclosure*, was introduced: the vendor is allowed some time to develop a patch before the vulnerability is disclosed by the finder [26]. This way, the vendor is given the opportunity to fix the vulnerability in time without the dangers of immediate exposure, but the public can still learn about the existence of the vulnerability in case the vendor fails to deliver a patch.

To coordinate the process between vendors, finders, and users, often a Computer Emergency Response Team/Coordination Center (CERT/CC) is used as a trusted third party. The economics of the vulnerability disclosure process has been studied [27] to shed light on the interaction between participating parties. There was a trend of using market strength to achieve the optimization of social benefit in the vulnerability disclosure process [28], though many [29, 30] criticize that the market solution performs no better than existing solutions.

Different vulnerability disclosure approaches have been compared and evaluated in their efficiency [31], and with the corresponding cost, benefits and risks provided in [32]. The work in [33] offers another angle for the life cycles of software vulnerabilities, which is based on analysis in seven dimensions for more than forty thousand vulnerabilities reported in the past few decades.

14.4 Automated Responsible Disclosure (ARD)

The parties to responsible disclosure, and the proposed *Automated Responsible Disclosure (ARD)* mechanism, can be divided into the following categories, as shown in Figure 14.1.:

- *Vendors* provide software and hardware-based (IoT) solutions to customers.
- *(Consortia of) Authorities* are in charge of managing security vulnerabilities; there may be multiple such authorities in a single country or region.
- *Security Experts* discover new vulnerabilities.
- *General public*, press, etc. who may be customers of the Vendors, or otherwise interested in the security of available services.

To improve responsible disclosure, the ARD solution has been designed with the following goals and assumptions:

1. The Security Experts cannot be intimidated to prevent the disclosure of vulnerabilities.
2. The Authorities can validate the reported vulnerabilities to prevent false reports.

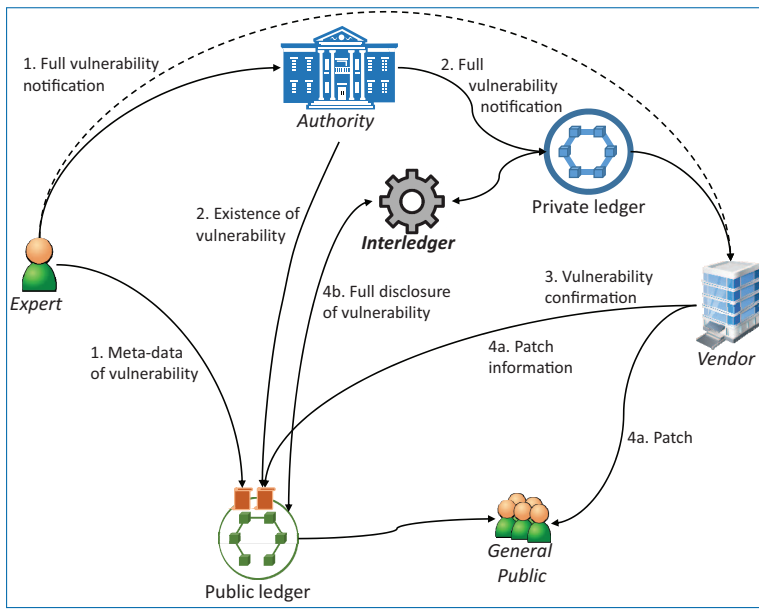


Figure 14.1. Entities of the interledger-enabled ARD solution.

3. The Vendor is clearly notified about each vulnerability and has to agree to fix it within the grace period; otherwise, the vulnerability is immediately disclosed.
4. Releasing a fix results in automatic disclosure of the vulnerability.
5. If a fix has not been released before the grace period has expired, the vulnerability is automatically disclosed.
6. Authorities are trusted to try to operate correctly, e.g., to prevent unintentional leaks of vulnerability information, correctly screen reported vulnerabilities, etc., but disclosure cannot be prevented by the Authority.

The solution is designed to utilize two distributed ledgers, a private one maintained by the Authority and used for storing the details of the vulnerability during the disclosure process, and a public one for first disclosing the existence of a vulnerability and later the details of the vulnerability. A key element is then the interledger functionality, which facilitates the automatic disclosure of the information from the private ledger to the public one once the conditions for disclosure have been met. In the basic case, the interledger functionality is run by the Authority or some other trusted party, but for additional security and resilience, the interledger functionality can be implemented in a distributed manner over a consortium of parties.

Technically, the public and private ledgers can utilize any suitable ledger technology (permissioned for the latter, and permissionless or permissioned for the former). The public ledger is readable by all, while some of the write operations are restricted. Anyone has a permission to store metadata of the vulnerability information (including its hash), which can be achieved, e.g., by a smart contract owned by the Authority. Additionally, write operations related to confirmation of vulnerability, notification of patching the vulnerability, and full disclosure of vulnerability information are limited to the Authority, Vendors, interledger, and the Security Expert who reported the vulnerability.

The private ledger can be implemented, for example, using Hyperledger Fabric, a permissioned DLTs, and a separate *chaincode* (which corresponds to a smart contract in Fabric) per Vendor. In Fabric, the vulnerability information is stored as a *private data collection* and the chaincode controls the access to the vulnerability using a hash-lock (see Siris *et al.* [10]). The hash-lock's secret is generated by the Security Expert and also known by both the Authority and Vendor and is revealed as necessary to the interledger, after which the interledger functionality can retrieve the vulnerability information from the private ledger and write it to the public ledger. The same secret is then used to lock the functionality Vendor is using for notifying of fixing the vulnerability, so Vendor's notification would reveal the secret, which would then allow the interledger functionality to retrieve and publish the full vulnerability information.

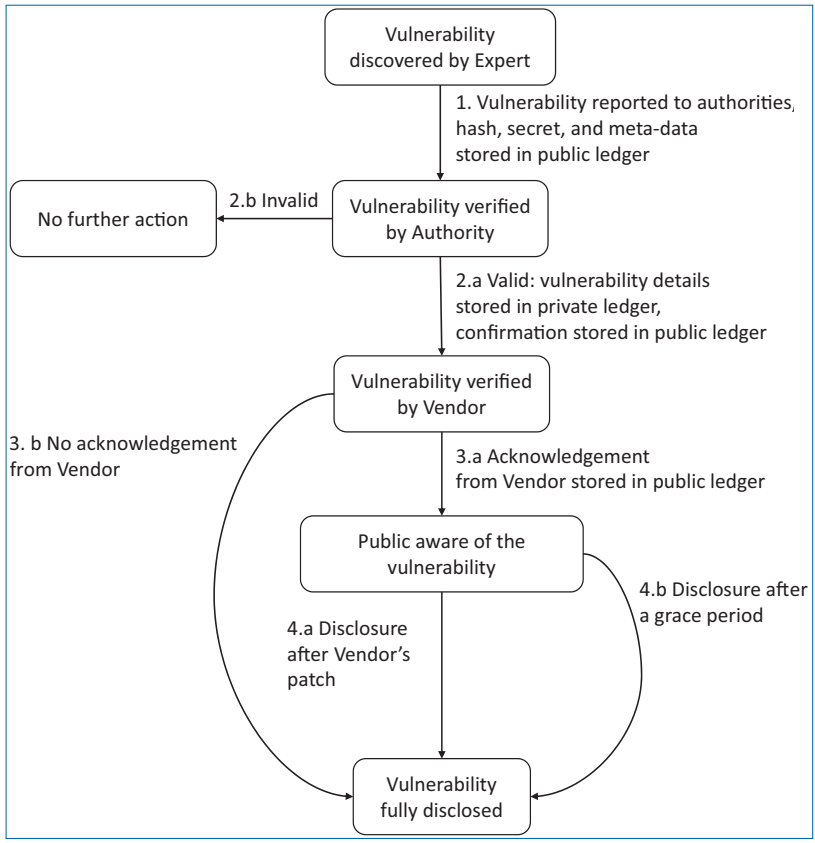

Vulnerability discovered by Expert
1. Vulnerability reported to authorities, hash, secret, and meta-data stored in public ledger
Vulnerability verified by Authority
2.b Invalid
No further action
2.a Valid: vulnerability details stored in private ledger, confirmation stored in public ledger
Vulnerability verified by Vendor
3. b No acknowledgement from Vendor
3.a Acknowledgement from Vendor stored in public ledger
Public aware of the vulnerability
4.b Disclosure after a grace period
4.a Disclosure after Vendor's patch
Vulnerability fully disclosed

Figure 14.2. State diagram of the ADR mechanism.

If there is a bounty mechanism for discovering vulnerabilities, the Security Experts can be identified using ephemeral decentralized identifiers (DIDs) or similar anonymous identifiers thus allowing them to claim the bounty while preventing the vendor from intimidating the Experts. The other parties in the solution can utilize longer term public identifiers including public DIDs.

ARD functions as follows, as described in Figure 14.2.:

1. The Security Expert, who has discovered a vulnerability V, generates the associated secret s and reports the vulnerability (together with the secret s) with a sufficient detail to the Authority, and optionally to the Vendor. The Security Expert also calculates the hash over the vulnerability information and stores this hash, $H(V)$, along with hash over the secret, $H(s)$, and other relevant metadata (Vendor's name, product name, affected version, etc.) to the public ledger, using a previously published mechanism, e.g., a smart contract.
2. The Authority, once they have ascertained the report is not bogus, stores the details of the vulnerability in the private ledger. The interledger functionality

then commits a transaction to public ledger stating that the Authority confirms that the previously reported vulnerability (with hash $H(V)$) is valid. If the report is a duplicate of a previous report, this is noted in the confirmation and the disclosure will follow the timeline of the original report.

3. The Vendor is notified of the new vulnerability. They now have a fixed time period (e.g., two weeks) to fetch the details of the vulnerability from the private ledger and either:
   a. confirm and agree to fix the vulnerability by storing their agreement to the public ledger, in which case they are accorded the grace period of, e.g., six months,[2] or
   b. deny/ignore the vulnerability and do nothing, in which case the Authority will reveal the secret s[3] to the public ledger and the interledger functionality will automatically disclose the vulnerability from the private ledger.
4. If the vendor has agreed to fix the vulnerability, and then
   a. releases a fix and stores a corresponding notice on the public ledger to indicate the vulnerability is now fixed, after which the interledger functionality will publish the full vulnerability information, which in turn will mark the vulnerability resolved. If the vendor wants its customers to have some extra time to update their systems, it can first indicate the fix is upcoming, and then after some time period marks the vulnerability as fixed. If, however the grace period runs out with no fix,
   b. the interledger functionality automatically discloses the vulnerability as above.
5. After the vulnerability has been disclosed, the security expert can claim the vulnerability report for credit and any bounty offered.

In Steps 1 and 2, the vulnerability report has to identify the Vendor (e.g., with their DID) and the product (e.g., with a vendor-specific identifier provided by the vendor). The same vendor and product ids are then included in the notice on the public ledger, which allows general public to know the number of known but not yet fixed vulnerabilities per vendor and per product.

In Step 3, if the vendor acknowledges the vulnerability, the vendor signs a proof of delivery (with their private key, tied to the public DID) containing the timestamp, the hash of vulnerability disclosure *H(V)*, and the relevant metadata (affected product name and version, etc.). This acknowledgement is stored in the publicly accessible ledger.

2. The Vendor can also mark the vulnerability as a duplicate of the previously reported one, in which case it will be disclosed simultaneously with the original vulnerability, as mentioned above.

3. If the Authority does not reveal the secret s in a timely manner, the Security Expert can also do the same.

Therefore, in the solution the vulnerability is always automatically disclosed by the interledger functionality in one of the three possible moments depending on the actions of the vendor:

- Immediately (if the vendor does not acknowledge it)
- After a fix has been released
- After the grace period if no fix has been released

The periods for acknowledging the vulnerability, after a fix has been released, and the grace period can be customized by the authority based on country, necessary security level, etc. For instance, for critical products, the period can be shorter.

Similarly, the bounty system can be customized: it can, e.g., contain a fixed compensation by the authority and an additional compensation by the vendor or by third parties for select products.

14.5 Analysis

The solution meets the five requirements listed in the beginning of Section 14.3:

1. *The security experts cannot be intimidated to prevent disclosure.*
   All vulnerabilities are reported under ephemeral anonymous identifiers thus maintaining the anonymity of the security expert throughout the process and preventing any intimidation.
2. *The authorities can validate the reported vulnerabilities to prevent false reports.*
   The authority vets all vulnerabilities before they are accepted to a suitable level to weed out bogus reports.
3. *The vendor is clearly notified about each vulnerability and has to agree to fix it to gain the grace period; otherwise, the vulnerability is immediately released.*
   The existence of a new vulnerability related to a product of the Vendor is clearly published on the public ledger to provide the vendor sufficient time to agree to or deny the vulnerability. If the Vendor does not agree to fix the vulnerability, the Authority publishes the secret s on the public ledger after which the vulnerability is automatically disclosed.
4. *Releasing a fix results in an automatic disclosure of the vulnerability.*
   Publishing the fix on the public ledger triggers the automatic disclosure after a short wait period.
5. *If a fix has not been released before the grace period has expired, the vulnerability is automatically disclosed.*
   The Authority (or the Security Expert) will reveal the secret s after the grace period has run out, after which the interledger functionality will automatically disclose the vulnerability.

To further evaluate the solution, the potential dishonesty of each party has to be considered.

If the Security Expert reports a non-valid vulnerability. There is the potential risk that the system is flooded with irrelevant information, e.g., to make the product appear to contain more vulnerabilities than it actually has. To prevent that, the Authority vets all reports, which can help prevent the obvious bogus reports, but non-trivial fakes may pass depending on the level of scrutiny. However, in this respect, the solution is not worse than existing solutions.

Duplicate reports of the same vulnerability. Valid but duplicate reports (either intentional or unintentional) of the same vulnerability can make the product appear of lower quality due to the increased number of reported vulnerabilities. To avoid this, duplicate reports can be marked by either the Authority or Vendor as duplicates of a previous report. Thus, they will not affect the number of individual vulnerabilities discovered, but each report will be eventually disclosed using the same timeline as the original report.

The Authority does not accept valid reports. This is always a potential issue as one can either have a mechanism to vet the reports to prevent spamming or have guaranteed acceptance of reports with a related spam issue. To reduce the risk of valid reports not being accepted, in addition to reporting the vulnerability to the Authority and optionally to the Vendor, the Security Expert stores the hash of vulnerability with associated metadata in the public ledger. If the Authority has not accepted a valid report, the Security Expert can then publicly reveal the vulnerability,[4] which together with the previously stored hash will make it obvious that the Authority did not handle the reported vulnerability properly.

The Authority does not disclose the vulnerability after the grace period has expired. In this case, both the Security Expert and general public will know that the vulnerability information has not been disclosed properly by the Authority, and the disclosure process can also be triggered by the Security Expert, who also possesses the secret needed.

The Vendor denies the existence of a valid vulnerability. In that case, the vulnerability is automatically disclosed thus providing a clear incentive to remain truthful or risk lose customers' trust.

The Vendor claims the fix is effective when it's not. Releasing a fix triggers a disclosure, which would reveal the fix is not effective thus providing a clear incentive to remain truthful.

4. This can be performed either using the same smart contract used for reporting vulnerability metadata, or any other channel.

The Vendor tries to extend the grace period. All time periods are automatically enforced preventing any extensions.

Furthermore, it is important to note that the actions performed by all parties, including the actions of the entity providing the interledger functionality, are transparently and immutably recorded on the public ledger. Hence, the transparency is increased and any attempts for misbehavior can be identified in a non-repudiable manner. The general public will also know the number of found, confirmed, but unpatched vulnerabilities per Vendor and product. This provides incentives for Vendors to patch vulnerabilities quickly and create more secure products.

14.6 Conclusions and Future Work

This chapter has introduced the Automated Responsible Disclosure (ARD) approach to automate the process of vulnerability disclosure for products and systems, designed a mechanism to implement it, and presented an initial analysis confirming the achievement of the design. This approach is motivated by, and addresses in particular, IoT systems risk management.

The solution is an interledger-enabled automated responsible disclosure providing accountability and appropriate incentives to the involved parties. The solution relies on DLTs, smart contracts and chaincode, interledger technologies, and decentralized identifiers, and extends the state of the art in responsible disclosure. The analysis of the design shows that the goals set are achieved automatically, even with external forces trying to disrupt the normal operations. Immutable recordings of actions of the involved parties on a public blockchain provide additional assurances (and incentives for the involved parties) in the case the design assumptions do not hold (or in case of various failures).

The designed mechanism does not include any mechanisms to automatically compensate the expert(s) who found and reported the vulnerability. However, DLTs and interledger mechanisms are very appropriate for providing this functionality as well. It was omitted from this discussion in order to simplify the design and avoid obscuring the key functionality and properties of the solution. This would be an expected extension after the approach has been accepted in the community, but it could also be an added feature and advantage for the approach to become accepted.

Further work is required in order to establish, promote, and practically evaluate the approach in the real world. Relevant authorities, responsible for public security and safety should probably push for the establishment of such a solution and the ecosystem around it.

Acknowledgments

The research reported here has been undertaken in the context of SOFIE (Secure Open Federation for Internet Everywhere) project, which has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No. 779984.

References

[1] J. Trull. Responsible Disclosure: Cyber Security Ethics. CSO Online, 2015. Available at: https://www.csoonline.com/article/2889357/responsible-disclosure-cyber-security-ethics.html (Accessed 5.2.2020).

[2] L. O'Donnell. 2 Million IoT Devices Vulnerable to Complete Takeover. Threatpost, 2019. Available at: https://threatpost.com/iot-devices-vulnerable-takeover/144167/ (Accessed 17.2.2020).

[3] D. Voolf, S. Boddy, and R. Cohen. Gafgyt Targeting Huawei and Asus Routers and Killing Off Rival IoT Botnets. F5 Labs, 2019. Available at: https://www.f5.com/labs/articles/threat-intelligence/gafgyt-targeting-huawei-and-asus-routers-and-killing-off-rival-iot-botnets (Accessed 17.2.2020).

[4] T. Spring. The Vulnerability Disclosure Process: Still Broken. Threatpost, 2018. Available at: https://threatpost.com/the-vulnerability-disclosure-process-still-broken/137180/ (Accessed 5.2.2020).

[5] K. Zetter. A Bizarre Twist in the Debate Over Vulnerability Disclosures. Wired, 2015. https://www.wired.com/2015/09/fireeye-enrw-injunction-bizarre-twist-in-the-debate-over-vulnerability-disclosures/ (Accessed 5.2.2020).

[6] C. Cimpanu. Researcher publishes second Steam zero day after getting banned on Valve's bug bounty program. ZDNet, 2019. https://www.zdnet.com/article/researcher-publishes-second-steam-zero-day-after-getting-banned-on-valves-bug-bounty-program/ (Accessed 5.2.2020).

[7] A. M. Algarni and Y. K. Malaiya. Software Vulnerability Markets: Discoverers and Buyers. Journal of Computer, Information Science and Engineering **8**, no. 3 (2014), 71–81.

[8] N. Stifter, A. Judmayer, P. Schindler, A. Zamyatin, and E. Weippl. Agreement with Satoshi – On the Formalization of Nakamoto Consensus. Cryptology ePrint Archive, Report 2018/400, 2018.

[9] N. Fotiou and G. C. Polyzos. Smart Contracts for the Internet of Things: Opportunities and Challenges. European Conference on Networks and Communications (EuCNC), Ljubljana, Slovenia, June 2018.

[10] V. A. Siris, P. Nikander, S. Voulgaris, N. Fotiou, D. Lagutin, and G. C. Polyzos. Interledger Approaches. IEEE Access **7** (2019), 89948–89966.
[11] C. Allen. The Path to Self-Sovereign Identity. April 2016. Available at: http://www.lifewithalacrity.com/2016/04/the-path-to-self-soverereignidentity.html (Accessed 18.12.2018).
[12] Blockchain and Identity: Projects/companies working on blockchain and identity. Available at: https://github.com/peacekeeper/blockchainidentity (Accessed 7.11.2018).
[13] D. Reed *et al.* Decentralized Identifiers (DIDs) v1.00 – Core Data Model and Syntaxes. W3C Working Draft 09 December 2019. Available at: https://www.w3.org/TR/did-core/ (Accessed 12.2.2020).
[14] Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation).
[15] H. Cavusoglu, H. Cavusoglu, and S. Raghunathan. Emerging Issues in Responsible Vulnerability Disclosure. WEIS. 2005.
[16] A. Arora, R. Telang, and H. Xu. Optimal policy for software vulnerability disclosure. Management Science **54**, no. 4, (2008), 642–656.
[17] L. J. Trautman and P. C. Ormerod. Industrial cyber vulnerabilities: Lessons from Stuxnet and the Internet of Things. U. Miami L. Rev. 72 (2017), 761.
[18] A. Nakajima, *et al.* A Pilot Study on Consumer IoT Device Vulnerability Disclosure and Patch Release in Japan and the United States. Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security. 2019.
[19] M. Stanislav and T. Beardsley. Hacking iot: A case study on baby monitor exposures and vulnerabilities. Rapid7 Report (2015).
[20] A. Arora and R. Telang. Economics of software vulnerability disclosure. IEEE Security & Privacy **3**, no. 1, (2005), 20–25.
[21] H. Cavusoglu and S. Raghunathan. Efficiency of Vulnerability Disclosure Mechanisms to Disseminate Vulnerability Knowledge. IEEE Transactions on Software Engineering **33**, no. 3, (2017), 171–185.
[22] J. Schiller. Responsible vulnerability disclosure: a hard problem. Secure Business Quarterly **2**, no. 1–5 (2002).
[23] M. McQueen, J. L. Wright, and L. Wellman. Are Vulnerability Disclosure Deadlines Justified? Third International Workshop on Security Measurements and Metrics, Banff, AB, (2011) 96–101.
[24] W. Pond. Do security holes demand full disclosure?. eWeek (2000).

[25] S. Ransbotham and S. Mitra. The impact of immediate disclosure on attack diffusion and volume. Economics of Information Security and Privacy III, Springer (2013), 1–12.
[26] A. Stone. Software flaws, to tell or not to tell?. IEEE Software **20**, no. 1, (2003), 70–73.
[27] J. Bollinger. "Economies of disclosure." ACM SIGCAS Computers and Society **34**, no. 3, (2004): 1–1.
[28] A. Ozment. Bug auctions: Vulnerability markets reconsidered. Third Workshop on the Economics of Information Security (2004).
[29] K. Kannan and Rahul Telang. Market for software vulnerabilities? Think again. Management Science **51**, no. 5, (2005), 726–740.
[30] S. Ransbotham, Sam, S. Mitra, and J. Ramsey. Are markets for vulnerabilities effective? Mis Quarterly (2012), 43–64.
[31] H. Cavusoglu, H. Cavusoglu and S. Raghunathan. Efficiency of Vulnerability Disclosure Mechanisms to Disseminate Vulnerability Knowledge. IEEE Transactions on Software Engineering, **33**, no. 3, (2007), 171–185.
[32] A. Cencini, K. Yu, and T. Chan. Software vulnerabilities: full-, responsible-, and non-disclosure. (2005). Available from: https://courses.cs.washington.edu/courses/csep590/05au/whitepaper_turnin/software_vulnerabilities_by_cencini_yu_chan.pdf (Accessed 17.2.2020).
[33] M. Shahzad, M. Z. Shafiq, and A. X. Liu. A large scale exploratory analysis of software vulnerability life cycles. Proceedings of the 34th International Conference on Software Engineering (ICSE), Zurich, (2012), 771–781.

Epilogue

The rise of the IoT paradigm provides enterprises with significant opportunities to implement novel systems that improve the automation and intelligence of their business processes. However, it also comes with a host of new cybersecurity challenges, which can compromise the ability of organizations to benefit from advanced IoT capabilities. Most of these challenges stem from the unique characteristics of IoT systems that differentiate them from other IT systems. Specifically, the cyber-physical nature of IoT systems obliges organizations to deal with OT (Operational Technology) and IT (Information Technology) security at the same time. Furthermore, IoT systems include new types of assets, such as IoT devices and cyber-physical systems, which introduce novel vulnerabilities and threats that are not covered by existing security knowledge bases. Likewise, industrial IoT systems (e.g., digital automation platforms in industrial plants) are increasingly becoming interconnected, which opens new opportunities for adversaries to attack industrial platforms. Moreover, organizations that deploy IoT must make sure that they comply with existing and emerging regulations, including general purpose regulations (such as the GDPR) and sector-specific regulations. In the light of these challenges, conventional cybersecurity methods must be enhanced with new tools for IoT systems and services.

In this book, we have introduced a range of novel IoT security systems and technologies that can cope with the above-listed challenges. The presented IoT security systems take advantage of leading-edge digital technologies in order to provide novel security tools and capabilities that are hardly possible based on conventional cybersecurity techniques. Prominent examples of such digital technologies include machine learning and blockchain technologies. Likewise, the novel techniques that are presented in the book include techniques for IoT data modeling, cybersecurity risk assessment, mapping of technical security measures to GDPR compliance

requirements, techniques for security information data sharing, security interoperability solutions, and more. Specifically, the following research methods and techniques are presented in the various chapters:

- Security data models for risk assessment in IoT systems, as well as for the configuration of the associated IoT security platforms and mechanisms, and ambient assisted living.
- Deep learning techniques based on Variational Autoencoders (VAE) for anomaly detection in IoT applications that comprise intelligent and semi-autonomous smart objects such as drones and robots.
- Privacy awareness techniques for IoT systems, along with methodologies and approaches for data protection that boost privacy compliance.
- Policy-based approaches for risk mitigation of IoT networks and devices, including approaches that leverage machine learning for detecting risks in real time.
- Architectures and platforms for end-to-end security, privacy, and safety in Industrial IoT systems, notably platforms that following popular edge/fog computing architectures for securing IoT systems. The presented platforms provide end-to-end security solutions, which address all types of IoT assets from the device up to the services and applications levels.
- Pattern-driven approaches for ensuring Security, Privacy, Dependability and Interoperability in IoT applications. The interoperability solutions include semantic interoperability for security data and mechanisms.
- Use of the LINDDUN privacy threat modeling methodology and of the STRIDE threat model towards evidence-based risk management and data protection, as a means of facilitating GDPR compliance.
- Approaches for cybersecurity certification, which combine certification with testing in order to alleviate the limitations of conventional certification tools for IoT environments (e.g., their inability to cope with dynamic and volatile environments).
- Device level security techniques, including techniques based on analysis of the source code and the binaries of the firmware of IoT nodes (e.g., gateways, sensing devices, smart objects).
- Decentralized (blockchain-based) solutions for boosting data legitimacy, privacy, and trust in industrial IoT infrastructures.
- Interledger technologies comprising multiple blockchain infrastructures towards federating different IoT platforms and performing combined risk assessments, i.e., risk assessments across different IoT platforms.

Most of the above-listed technologies and techniques are innovative and hold the promise to increase the security and resilience of next generation large-scale IoT systems, including systems that will comprise smart objects. The chapters of the book present how these security techniques are used to protect IoT systems and applications in a variety of sectors including, for example, transport, healthcare, and industry. These applications are in several cases indicative of the future IoT environments and their security challenges.

The solutions of the book are collectively presenting some of the main characteristics of future IoT security environments. The latter are expected to play an important role for the rapid adoption and wider use of IoT systems in the scope of the fourth industrial revolution (Industry 4.0). The wave of Industry 4.0 system will comprise large number of IoT devices, including different IoT assets (e.g., smart objects, IoT services, next generation networks (e.g., 5G)) in complex and dynamic configurations. The solutions listed in the book could greatly boost IoT adoption in the Industry 4.0 era, through lowering barriers associated with security, privacy and data protection.

The solutions and technologies of the book will play a significant role in addressing security, privacy, and data protection in future IoT systems, notably systems that are characterized by semi-autonomous behaviors, integration between physical and cybersecurity, expanded use of AI as part of IoT functionalities, as well as richer and more frequent security information sharing across IoT stakeholders. Even though it is not easy to predict the future, we can safely assume that IoT security will be a challenging, yet very exciting journey for security experts, consultants, security research organizations, IIoT solution providers and many other stakeholders. In this book, we have provided knowledge and insights about where we stand in this journey. We also attempted to develop a vision for the future. We really hope you will enjoy the journey and will appreciate our ideas to help you start it on the right foot.

Index

anomaly detection, xvii, 51–53, 55, 59, 61, 90, 92, 96–99, 248
autoencoders, 55, 57–59, 65, 92, 248
big data, 21, 208
CAPEC, 9, 25, 33
CERTs, 9, 37, 175
CMDB, 8, 25, 29, 33, 35, 39, 42, 43
cognitive packet network, xix, 105, 116, 119
CPE, 9, 25, 33
CTI, 9, 19, 25
CVE, 9, 19, 25, 33, 35
CWE, 9, 25, 33
cyber physical systems (CPS), xvi, 2, 7, 13, 18, 35, 129, 145, 209, 210, 247
data sets, xvii, 13, 49, 51–55, 57–66, 96, 131
deep learning, xvii, 13, 48, 49, 51, 52, 55, 57, 59, 65, 66, 90, 96, 248
dependability, xviii, 121, 123, 125, 126, 128, 130, 135, 137, 248
design patterns, 125, 128, 132
digital twins for security, 247
distributed ledgers, xix, 238
GDPR, xvi–xviii, 5, 12–14, 18, 26, 69–79, 81, 84, 85, 124, 144, 146, 153, 157, 162–166, 168–173, 175, 177, 183, 235, 247, 248
IIC, 10, 18, 26
IISF, 10, 18, 50
interledger, xix, 229, 232–234, 237–241, 243, 248
Internet of Things (IoT), xv–xx, 1–3, 5–10, 12–14, 17–31, 34, 35, 37–39, 41, 42, 46, 47, 49–53, 58, 65, 66, 69–81, 84, 85, 88, 108, 114, 115, 121–131, 133, 137, 143, 144, 149, 156, 157, 161, 164, 167, 163, 177–184, 186, 189, 192, 196–198, 202, 203, 205, 208–218, 220, 221, 225, 226, 229, 230, 232, 234–237, 243, 247–249
interoperability, xvi, xviii, 19, 25, 28, 85, 121–123, 127, 128, 130, 137, 223, 232, 248
IoT security, xvi, xvii, xix, xx, 9, 124, 183, 196
JSON, 9, 44, 176, 224
machine learning, xvi, xviii, xx, 9, 13, 19, 24, 49, 51, 53, 88, 90, 92, 99, 105, 109, 117, 119, 247, 248
NFV, 123, 136

OVAL, 10
OWASP, 9, 175
privacy, xvi–xx, 1, 11–13, 22, 25, 26, 69–76, 78–85, 105, 106, 110–113, 117, 119, 120, 122–125, 128–130, 133–135, 137, 143–149, 153, 157, 158, 162, 164–167, 175–177, 179, 183, 209, 210, 223, 234, 235, 248, 249
random neural networks, 52
responsible disclosure, 229–232, 236, 237, 243
risk assessment, xvi–xviii, xx, 1, 5–10, 13, 14, 17, 19–29, 33, 34, 36–41, 44–47, 79, 89–91, 144–146, 148, 157, 170, 171, 180, 184–186, 190, 192, 247, 248
SDN, 92, 93, 99, 122, 131–133, 136
security, xvi, xvii, xix, xx, 1–3, 5, 7–15, 17–30, 33, 34, 37, 46–51, 70, 72, 77, 80, 88–96, 99, 106, 115–117, 119, 122, 124, 126, 128, 129, 133–135, 144–148, 153, 157, 158, 162, 163, 170, 173, 174, 178–186, 188, 190–192, 196–199, 201–206, 208–221, 223, 224, 229–232, 235, 237, 238, 240, 241, 247–249
security certification, xvi, xix, 178–184, 186, 191, 192
security challenges, xvii, 2, 209, 249
security knowledge base, 8, 9, 14, 25, 26, 28, 33, 37, 247
security modeling, xvii, 29, 30, 32, 38
security risk assessment, 7, 13, 184, 185, 190
security testing, 180, 183–185, 188, 190
security vulnerabilities, 5, 95, 106, 197, 199, 202, 205, 209, 211, 229, 230, 235, 237
software defined networks, 90
standardization, 70, 71, 73, 75, 76, 93, 179, 180
XML, 9, 19, 20, 28, 47, 176, 189

About the Editor

John Soldatos (http://gr.linkedin.com/in/johnsoldatos) holds a PhD in Electrical & Computer Engineering from the National Technical University of Athens (2000); was Associate Professor and Head of the Internet of Things (IoT) Group at the Athens Information Technology (AIT), Greece (2006–2019); and was Adjunct Professor at the Carnegie Mellon University, Pittsburgh, PA (2007–2010). Since January 2020, he joined INTRASOFT International as part of the incorporation of AIT's IoT Group in the company. He is also Honorary Research Fellow at the University of Glasgow, UK (2014-present). Dr. Soldatos is expert in IoT technologies and applications, including applications in smart cities and the fourth industrial revolution (Industry 4.0). Dr. Soldatos has played a leading role in the successful delivery of more than sixty (commercial-industrial, R&D, and consulting) projects, for both private & public sector organizations, including some complex integrated projects. He is co-founder of various platforms and initiatives such as the open source OpenIoT platform (https://github.com/OpenIotOrg/openiot) and the Edge4Industry (www.edge4industry.eu) community. He has published more than 180 articles in international journals, books, and conference proceedings. He has also significant academic teaching experience, along with experience in executive education and corporate training. Dr. Soldatos has been regular contributor and columnist in various international magazines and blogs about IoT/Industry 4.0 technologies and applications. He has also received national and international recognition through appointments in standardization working groups, expert groups, and various boards. Moreover, he has been Scientific Advisor to several high-tech start-up enterprises. He has also co-edited and co-authored two edited volumes (books) on Internet of Things topics, including one on IoT for Industrial Automation and another on IoT Analytics.

Contributing Authors

Stefanos Astaras
Athens Information Technology,
Kifisias Ave. 44, Marousi, GR15125
sast@ait.gr

Alice Audino
Istituto Italiano per la Privacy
a.audino@istitutoprivacy.it

Pasquale Annicchino
Archimede Solutions
pannicchino@archimede.ch

Gianmarco Baldini
European Commission, Joint Research Centre (JRC), Ispra 21027, Italy
gianmarco.baldini@ec.europa.eu

Andrea Battaglia
ASPISEC SRL, Rome, Italy

Luca Bolognini
Istituto Italiano per la Privacy
l.bolognini@istitutoprivacy.it

Arne Bröring
Siemens AG Corporate Technology
Siemens, Munich, Germany
arne.broering@siemens.com

Paul-Emmanuel Brun
AIRBUS CyberSecurity
paul-emmanuel.brun@airbus.com

Bora Caglayan
IBM, Dublin, Ireland

Davide Conzon
LINKS Foundation – Leading Innovation & Knowledge for Society, Turin, Italy
davide.conzon@linksfoundation.com

Stelios Christofi
EBOS Technologies, Cyprus

Francesco Capparelli
Istituto Italiano per la Privacy
f.capparelli@istitutoprivacy.it

Jacek Dominiak
Beawre Digital SL

Anastasis Drosou
Centre for Research and Technology Hellas, Information Technologies Institute (ITI), Greece
drosou@iti.gr

Angela-Maria Despotopoulou
Intrasoft International, 19.5 Km. Markopoulou Ave., GR19002
AngelaMaria.Despotopoulou@intrasoft-intl.com

Spyridon Evangelatos
Intrasoft International, 19.5 Km. Markopoulou Ave., GR19002
Spyros.Evangelatos@intrasoft-intl.com

Enrico Ferrera
LINKS Foundation – Leading Innovation & Knowledge for Society, Turin, Italy
enrico.ferrera@linksfoundation.com

Nikos Fotiou
Mobile Multimedia Laboratory, Department of Informatics, School of Information Sciences and Technology, Athens University of Economics and Business, Greece
fotiou@aueb.gr

Piotr Fröhlich
Polish Academy of Sciences, Institute of Theoretical and Applied Informatics
pfrohlich@iitis.pl

Konstantinos Fysarakis
Sphynx Technology Solutions AG, Zug, Switzerland
fysarakis@sphynx.ch

Erol Gelenbe
Polish Academy of Sciences, Institute of Theoretical and Applied Informatics
e.gelenbe@imperial.ac.uk

Elena González
Beawre Digital SL

Jose Luis Hernandez-Ramos
European Commission, Joint Research Centre (JRC), Ispra 21027, Italy
jose-luis.hernandez-ramos@ec.europa.eu

Sotiris Ioannidis
Institute of Computer Science, Foundation for Research and Technology-Hellas (FORTH-ICS), Heraklion, Greece
sotiris@ics.forth.gr

Yki Kortesniemi
Department of Communications and Networking, Aalto University, Espoo, Finland

Department of Computer Science, Aalto University, Espoo, Finland
yki.kortesniemi@aalto.fi

Thomas Krousarlis
INLECOM Innovation, Athens, Greece
thomas.krousarlis@inlecomsystems.com

Florent Kirchner
CEA LIST, Software Reliability and Security Laboratory, Palaiseau, France

Vivek Kulkarni
Siemens AG Corporate Technology Siemens, Munich, Germany
vivekkulkarni@siemens.com

Nikos Kefalakis
Intrasoft International, 19.5 Km. Markopoulou Ave., GR19002
Nikos.Kefalakis@intrasoft-intl.com

Dmitrij Lagutin
Department of Communications and Networking, Aalto University, Espoo, Finland
dmitrij.lagutin@aalto.fi

Konstantinos Loupos
INLECOM Innovation, Athens, Greece
konstantinos.loupos@inlecomsystems.com

Sara N. Matheu
Department of Information and Communications Engineering, University of Murcia, Spain

Guillemette Massot
AIRBUS CyberSecurity
guillemette.massot@airbus.com

Manolis Michalodimitrakis
Institute of Computer Science, Foundation for Research and Technology-Hellas (FORTH-ICS), Heraklion, Greece
manmix@ics.forth.gr

Victor Muntés-Mulero
Beawre Digital SL
victor.muntes@beawre.com

Antonis Mygiakis
INLECOM Innovation, Athens, Greece
antonis.mygiakis@inlecomsystems.com

Mateusz Nowak
Polish Academy of Sciences, Institute of Theoretical and Applied Informatics
mateusz@iitis.pl

Slawek Nowak
Polish Academy of Sciences, Institute of Theoretical and Applied Informatics
snowak@iitis.pl

Stavros Papadopoulos
Centre for Research and Technology Hellas, Information Technologies Institute (ITI), Greece
spap@iti.gr

Alexandros Papageorgiou
INLECOM Systems, London, UK
alex.papageorgiou@inlecomsystems.com

Manos Papoutsakis
Institute of Computer Science, Foundation for Research and Technology-Hellas (FORTH-ICS), Heraklion, Greece
paputsak@ics.forth.gr

Nikolaos Petroulakis
Institute of Computer Science, Foundation for Research and Technology-Hellas (FORTH-ICS), Heraklion, Greece
npetro@ics.forth.gr

Henrich C. Pöhls
PIDS – Passau Institute of Digital Security, Chair of IT-Security, University of Passau, Passau, Germany
hp@sec.uni-passau.de

George C. Polyzos
Mobile Multimedia Laboratory, Department of Informatics, School of Information Sciences and Technology, Athens University of Economics and Business, Greece
polyzos@aueb.gr

Mohammad Rifat Ahmmad Rashid
LINKS Foundation – Leading Innovation & Knowledge for Society, Turin, Italy rifat.abir@gmail.com

David Sanchez-Charles
Ludium Lab SL
david.sanchez@ludiumlab.com

Vasilios A. Siris
Mobile Multimedia Laboratory, Department of Informatics, School of Information Sciences and Technology, Athens University of Economics and Business, Greece
vsiris@aueb.gr

Antonio F. Skarmeta
Department of Information and Communications Engineering, University of Murcia, Spain

Christos Skoufis
EBOS Technologies, Cyprus

John Soldatos
Intrasoft International, 19.5 Km. Markopoulou Ave., GR19002
John.Soldatos@intrasoft-intl.com

George Spanoudakis
Sphynx Technology Solutions AG, Zug, Switzerland
spanoudakis@sphynx.ch

Basile Starynkevitch
CEA LIST, Software Reliability and Security Laboratory, Palaiseau, France

Xu Tao
LINKS Foundation – Leading Innovation & Knowledge for Society, Turin, Italy
xu.tao@linksfoundation.com

George Theofilis
CLMS Hellas, Athens, Greece

Dimitrios Tzovaras
Centre for Research and Technology Hellas, Information Technologies Institute (ITI), Greece
dimitrios.tzovaras@iti.gr

Aydin Ulas
IBM, Dublin, Ireland

Franck Vedrine
CEA LIST, Software Reliability and Security Laboratory, Palaiseau, France

Mario Villiani
ASPISEC SRL, Rome, Italy

Lei Wu
Department of Communications and Networking, Aalto University, Espoo, Finland
lei.1.wu@aalto.fi

Sofiane Zemouri
IBM, Dublin, Ireland

Sébastien Ziegler
Mandat International
sziegler@mandint.org